On particle filters applied to electricity load forecasting

Tristan Launay, Anne Philippe, Sophie Lamarche

On particle filters applied to electricity load forecasting

Tristan Launay[1,2] Anne Philippe[1] Sophie Lamarche[2]

April 15, 2013

Abstract

In this paper, we are interested in the online prediction of the electricity load, within the Bayesian framework of dynamic models. We offer a review of sequential Monte Carlo methods, and provide the calculations needed for the derivation of so-called particles filters. We also discuss the practical issues arising from their use, and some of the variants proposed in the literature to deal with them, giving detailed algorithms whenever possible for an easy implementation. We propose an additional step to help make basic particle filters more robust with regard to outlying observations. Finally we use such a particle filter to estimate a state-space model that includes exogenous variables in order to forecast the electricity load for the customers of the French electricity company Électricité de France and discuss the various results obtained.

Keywords: dynamic model, particle filter, sequential Monte Carlo, electricity load forecasting

1 Introduction

Let $\{X_n\}_{n\geq 0}$ and $\{Y_n\}_{n\geq 0}$ be $\mathcal{X} \subset \mathbb{R}^{n_x}$ and $\mathcal{Y} \subset \mathbb{R}^{n_y}$-valued stochastic processes defined on a measurable space. The observations $\{Y_n\}_{n\geq 0}$ are assumed conditionally independent given the hidden Markov process $\{X_n\}_{n\geq 0}$ most often referred to as the states of the model, and are characterised by the conditional density $g_n^\theta(y_n|x_n)$. We denote the initial density of the state as $\mu^\theta(x_0)$ and the Markov transition density from time $n-1$ to time n as $f_n^\theta(x_n|x_{n-1})$. The superscript θ on these densities is the parameter of the model, that belongs to an open set $\Theta \subset \mathbb{R}^{n_\theta}$. The model can be summarised (using practical and common if not exactly rigorous notations) as

$$X_0 \sim \mu^\theta(\cdot), \quad X_n|(X_{n-1} = x_{n-1}) \sim f_n^\theta(\cdot|x_{n-1}) \tag{1.1}$$

$$Y_n|(X_n = x_n) \sim g_n^\theta(\cdot|x_n). \tag{1.2}$$

Within the Bayesian framework, equations (1.1) specify the prior on the states of the model whose likelihood is defined via (1.2).

Notice here that we restrict ourselves to models with independent observations, but that the framework can easily be extended to include dependent observations if need be. The class of dynamic models we consider, known as general state-space models or hidden Markov models (HMM) in the literature and whose typical representation is given in Figure 1, includes many non linear and non Gaussian time series models such as

$$X_{n+1} = F_n(X_n, V_{n+1}) \tag{1.3}$$

$$Y_n = G_n(X_n, W_n) \tag{1.4}$$

[1]Laboratoire de Mathématiques Jean Leray, 2 Rue de la Houssinière – BP 92208, 44322 Nantes Cedex 3, France
[2]Electricité de France R&D, 1 Avenue du Général de Gaulle, 92141 Clamart Cedex, France

where $\{V_n\}_{n\geq 1}$ and $\{W_n\}_{n\geq 0}$ are independent sequences of independent random variables and $\{F_n\}_{n\geq 1}$ and $\{G_n\}_{n\geq 1}$ are sequences of (possibly non linear) functions. Such models find applications in many fields including time-series forecasting (Dordonnat, 2009), biostatistics (Rossi, 2004; Vavoulis et al., 2012), econometrics (Liu and West, 2001; Johansen et al., 2008; Chopin et al., 2012), telecommunications (Lee et al., 2010), object tracking (Rui and Chen, 2001; Gilks and Berzuini, 2001; Gustafsson et al., 2002; Karlsson, 2005), etc.

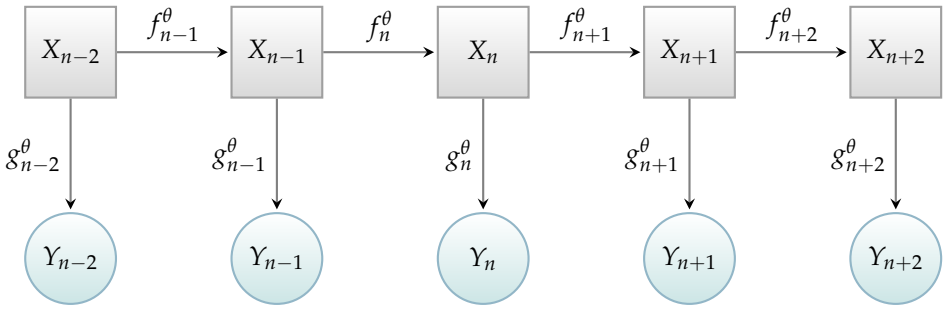

Figure 1: A generic hidden Markov Model (HMM).

When the parameter θ is known, on-line inference about the state process given the observations is a so-called optimal filtering problem. For simple models such as the linear Gaussian state-space model the problem can be solved exactly using the standard Kalman filter (see for example Durbin and Koopman, 2001), and the case of a finite state-space also allows for explicit calculations. For non linear models, the Extended Kalman filter is often used and relies on the approximation of the first derivative of F_n, although good performances are not guaranteed theoretically. Another technique is the so-called Unscented Kalman filter (see Wan and van der Merwe, 2000, for the comprehensive details) which makes use of the unscented transformation to deal with the non linearity of the system.

For our application, we are interested in the on-line prediction of the french electricity load through the estimation (and prediction) of a dynamic model and choose to consider Sequential Monte Carlo (SMC) methods also known as particle methods instead. SMC methods are a class of sequential simulation-based algorithms which aim at approximating the posterior distributions of interest. They represent a popular alternative to Kalman filters (Kantas et al., 2009) since they are often easy to implement, apply to non linear non Gaussian models, and have been demonstrated to yield accurate estimates (Doucet et al., 2001; Liu, 2008).

In Section 2 we introduce the key concepts behind sequential Monte Carlo methods. In Section 3 we first derive the algorithm for a basic particle filter and discuss common practical issues. We then review the main techniques appearing in the literature to deal with these issues and we also propose a new additional step to help make particle filters more robust with regard to outlying observations. Finally, we propose a new nonlinear dynamic model for the electricity load in Section 4 and use a particle filter to estimate this model. We compare the predictions we obtain to operational predictions and show that our model remains competitive, even though its definition is simpler than that of the model studied in Dordonnat et al. (2008).

2 Inference in hidden Markov models

Let us first assume that the parameter θ is known: the model with θ unknown will be discussed later in Section 3.7. Given equations (1.1) and (1.2), the posterior distribution of the states given

the observations is

$$\pi^\theta(x_{0:n}|y_{0:n}) \propto \underbrace{\prod_{k=1}^{n} g_k^\theta(y_k|x_k)}_{n \text{ likelihoods}} \cdot \underbrace{\prod_{k=1}^{n} f_k^\theta(x_k|x_{k-1})}_{n \text{ transition densities}} \cdot \underbrace{\mu^\theta(x_0)}_{\text{initial density}} . \tag{2.1}$$

From equation (2.1), three distinct goals might be pursued (see for example Chen, 2003; Cappé et al., 2010)

Filtering: the aim of filtering is to estimate the distribution of the state X_n conditionally to the observations up to time n, i.e. $y_{0:n}$.

Smoothing: the aim of smoothing is to estimate the distribution of the state X_n conditionally to the observations up to time n' (with $n' \geq n$), i.e. $y_{0:n'}$. Note that $\pi^\theta(x_n|y_{0:n})$ is both a filtered and a smoothed distribution.

Predicting: the aim of predicting is to estimate the distribution of the state $X_{n+\tau}$ (with an horizon $\tau > 0$) conditionally to the observations up to time n, i.e. $y_{0:n}$. From there, using (1.2), it is easy to forecast the upcoming observation $Y_{n+\tau}$ which is usually the real target. When not explicitly mentioned, the horizon considered for prediction will be $\tau = 1$.

To summarise, given the available observations, filtering focuses on the current state, smoothing focuses on the past states, and predicting focuses on the future states. Our goal being the online prediction of the electricity load, we chose to focus on predicting and filtering, since the filtered distribution of the state at time n is needed to produce forecasts for time $n + \tau$: ultimately, smoothing only refines the estimation of past states over time, without influencing the quality of the online prediction, and is therefore not needed to achieve our goal.

Markov Chains Monte Carlo

MCMC methods (see for example Robert, 1996; Robert and Casella, 2004; Marin and Robert, 2007) certainly represent a viable estimation procedure: most of the time, nothing really prevents the exploration via MCMC of the posterior distribution derived in (2.1) from the prior and the likelihood given in (1.1) and (1.2). From a practical point of view however, MCMC methods are most likely not the optimal tool: the addition of a new observation y_{n+1} from the model forces the overall re-estimation of the smoothed distribution of the states $\pi^\theta(x_{0:n+1}|y_{0:n+1})$ even when we are interested only in the last marginal of this distribution i.e. the filtered distribution $\pi^\theta(x_{n+1}|y_{0:n+1})$. The MCMC estimation is thus not recursive (with regard to the time index) in the sense that the filtered distribution $\pi^\theta(x_{n+1}|y_{0:n+1})$ at time $n + 1$ cannot be computed from the previous filtered distribution $\pi^\theta(x_n|y_{0:n})$ at time n using MCMC methods, which is a major drawback given the computationally expensive nature of these methods.

Notice also that even though designing the MCMC algorithm can be simple in some cases, the dimension of the space explored grows linearly with the time index making the assessment of the convergence of the produced Markov chains all the more complicated.

Importance sampling

Monte Carlo integration allows the estimation of integrals of the form

$$I = \mathbb{E}^\pi[h(X)] = \int h(x)\pi(x)\,\mathrm{d}x, \tag{2.2}$$

where π is a probability density and where $h \in \mathrm{L}^1(\pi)$. This method is often used to numerically approximate the expectation of a random variable whose density is π or a moment of higher order.

Let us assume that a probability density q (the so-called importance density) is available from which we can simulate, and such that the support of π is included in that of q. We can then write

$$I = \int h(x)\pi(x)\,\mathrm{d}x = \frac{\displaystyle\int \frac{h(x)\pi(x)}{q(x)} q(x)\,\mathrm{d}x}{\displaystyle\int \frac{\pi(x)}{q(x)} q(x)\,\mathrm{d}x}.$$

Given $X^1, \ldots, X^M$ i.i.d. random variables with probability density q, the self-normalised importance sampling estimator of I is defined by

$$\widehat{I}_M(q) = \frac{\displaystyle\sum_{j=1}^{M} \frac{h(X^j)\pi(X^j)}{q(X^j)}}{\displaystyle\sum_{j=1}^{M} \frac{\pi(X^j)}{q(X^j)}} = \sum_{j=1}^{M} w^j h(X^j), \tag{2.3}$$

where we define the self-normalised weights as

$$w^j = \frac{\widetilde{w}^j}{\sum_{k=1}^{M} \widetilde{w}^k}. \tag{2.4}$$

with

$$\widetilde{w}^j = \frac{\pi(X^j)}{q(X^j)}. \tag{2.5}$$

See Geweke (1989) for the theoretical details (including proof of the consistency of the estimator).

Sequential Monte Carlo

SMC methods provide a viable and popular alternative to MCMC methods for the Bayesian online estimation of dynamic models. Particle methods are recursive by nature (thus computationally cheaper than MCMC) and similar in some ways to the Kalman filter approach. Particle methods essentially draw their strength from the immediate calculations that we show below

$$\begin{aligned}
\pi^\theta(x_{0:n}|y_{0:n}) &= \frac{\pi^\theta(y_{0:n}|x_{0:n})\pi^\theta(x_{0:n})}{\pi^\theta(y_{0:n})} = \frac{\pi^\theta(y_n, y_{0:n-1}|x_{0:n})\pi^\theta(x_{0:n})}{\pi^\theta(y_n, y_{0:n-1})} \\
&= \frac{\pi^\theta(y_n|y_{0:n-1}, x_{0:n})\pi^\theta(y_{0:n-1}|x_{0:n})\pi^\theta(x_{0:n})}{\pi^\theta(y_n|y_{0:n-1})\pi^\theta(y_{0:n-1})} \\
&= \frac{\pi^\theta(y_n|y_{0:n-1}, x_{0:n})\pi^\theta(x_{0:n}|y_{0:n-1})\pi^\theta(y_{0:n-1})\pi^\theta(x_{0:n})}{\pi^\theta(y_n|y_{0:n-1})\pi^\theta(y_{0:n-1})\pi^\theta(x_{0:n})} \\
&= \frac{\pi^\theta(y_n|x_{0:n})\pi^\theta(x_n|x_{0:n-1}, y_{0:n-1})}{\pi^\theta(y_n|y_{0:n-1})} \cdot \pi^\theta(x_{0:n-1}|y_{0:n-1})
\end{aligned}$$

i.e. with the notations we introduced earlier:

$$\begin{aligned}
\pi^\theta(x_{0:n}|y_{0:n}) &= \frac{g_n^\theta(y_n|x_n) f_n^\theta(x_n|x_{n-1})}{\pi^\theta(y_n|y_{0:n-1})} \cdot \pi^\theta(x_{0:n-1}|y_{0:n-1}) \\
&\propto g_n^\theta(y_n|x_n) f_n^\theta(x_n|x_{n-1}) \cdot \pi^\theta(x_{0:n-1}|y_{0:n-1}).
\end{aligned} \tag{2.6}$$

The recursive equation (2.6) plays a central role in the definition of all particle methods. An integrated version of this equation is most often presented to emphasise the direct connection

between two consecutive filtered distributions:

$$\begin{aligned}\pi^\theta(x_n|y_{0:n}) &= \int \pi^\theta(x_{0:n}|y_{0:n})\,\mathrm{d}x_{0:n-1} \\ &\propto g_n^\theta(y_n|x_n)\int f_n^\theta(x_n|x_{n-1})\cdot\pi^\theta(x_{n-1}|y_{0:n-1})\,\mathrm{d}x_{n-1}.\end{aligned} \tag{2.7}$$

The main idea behind particle filters is to make extensive use of equation (2.6) to compute sequential Monte Carlo approximations of the posterior distributions of interest, in our case, the sequence of filtered distributions. The general procedure is simple enough and mimics the iterative prediction-correction structure of any Kalman filter. A each time n the filtered density $\pi^\theta(x_n|y_{0:n})$ can be approximated by the empirical distribution of a large sample of M ($M >> 1$) weighted random samples termed particles. The weighted particles evolve over time: they follow the prior dynamic distribution of the model and get re-adjusted as soon as observations become available. At time n, the two basic steps (a lot of refinements are possible that we will discuss later on) of particle filters are the following:

Prediction: given particles distributed along density $\pi^\theta(x_{n-1}|y_{0:n-1})$, we simulate new particles distributed along density $\pi^\theta(x_n|y_{0:n-1})$ with the help of the transition density $f_n^\theta(x_n|x_{n-1})$.

Correction: we re-weight these particles distributed along density $\pi^\theta(x_n|y_{0:n-1})$ depending on the observation y_n with the help of (2.6) to approximate the distribution $\pi(x_n|y_{0:n})$.

Particle filters essentially combine Monte Carlo integration and importance sampling. We describe the application of self-normalised importance sampling to estimate a sequence of integrals that involve the posterior distribution (2.1) and that are of the form

$$\begin{aligned}I_n &= \int h(x_n)\pi^\theta(x_{0:n}|y_{0:n})\,\mathrm{d}x_{0:n} \\ &= \int h(x_n)\pi^\theta(x_n|y_{0:n})\,\mathrm{d}x_n.\end{aligned}$$

We use the self-normalised importance sampling estimator defined in (2.3), with $\pi(x) = \pi^\theta(x_{0:n}|y_{0:n})$ and $q(x) = q(x_{0:n}|y_{0:n})$. Given M particles $X_{0:n}^1, \ldots, X_{0:n}^M$, i.i.d. with probability density $q^\theta(x_{0:n}|y_{0:n})$, we will approximate I_n by

$$\widehat{I}_{n,M}^{\mathrm{PF}} = \sum_{j=1}^M w_n^j h(X_n^j),$$

where mimicking the definitions given (2.5) and (2.4) we define

$$w_n^j = \frac{\widetilde{w}_n^j}{\sum_{k=1}^M \widetilde{w}_n^k}, \tag{2.8}$$

with

$$\widetilde{w}_n^j = \frac{\pi^\theta(X_{0:n}^j|y_{0:n})}{q^\theta(X_{0:n}^j|y_{0:n})}. \tag{2.9}$$

Note that to alleviate the notational burden, we voluntarily omit the dependence of the importance weights on the parameter θ, and will do so for the remainder of the chapter when no confusion is possible.

A convenient form of importance density

Let us consider an importance density q that can be factorised as follows:

$$\begin{aligned} q^\theta(x_{0:n}|y_{0:n}) &= q^\theta(x_n|y_{0:n-1}, x_{0:n}) q^\theta(x_{0:n-1}|y_{0:n-1}) \\ &= q^\theta(x_0|y_0) \prod_{k=1}^{n} q^\theta(x_k|y_{0:k-1}, y_{0:k}). \end{aligned} \tag{2.10}$$

It is now easy to see, using (2.6), that the weights $\widetilde{w}_n^\theta(X_{0:n}^j)$ can be updated recursively via

$$\begin{aligned} \widetilde{w}_n^j &= \frac{\pi^\theta(X_{0:n}^j|y_{0:n})}{q^\theta(X_{0:n}^j|y_{0:n})} = \frac{g_n^\theta(y_n|X_n^j) f_n^\theta(X_n^j|X_{n-1}^j) \pi^\theta(X_{0:n-1}^j|y_{0:n-1})}{\pi^\theta(y_n|y_{0:n-1}) q^\theta(X_n^j|X_{0:n-1}^j, y_{0:n}) q^\theta(X_{0:n-1}^j|y_{0:n-1})} \\ &= \widetilde{w}_{n-1}^j \frac{g_n^\theta(y_n|X_n^j) f_n^\theta(X_n^j|X_{n-1}^j)}{\pi^\theta(y_n|y_{0:n-1}) q^\theta(X_n^j|X_{0:n-1}^j, y_{0:n})}. \end{aligned} \tag{2.11}$$

where $\pi^\theta(y_n|y_{0:n-1})$ does not depend on the index j, and need not be computed at all since the weights w_n^j featured in the estimator are the self-normalised version of the weights $\widetilde{w}_n^j$ (the constant vanishes after the self-normalisation). Note that w_{n-1}^j can be substituted to $\widetilde{w}_{n-1}^j$ in the recursive update (2.11) for the very same reason.

Equation (2.11) lies at the very core of all the particle filters in general, some variants of which we describe in the next section. It summarises, by itself, the edge that SMC methods have over MCMC methods in general in the context of dynamic models: it allows for sequential recursive estimations and predictions. At each time step, two things only are required to estimate the quantity of interest: simulations from the importance density q^θ (the choice of which shall be discussed) and the update of the particles' weights via the computation of (2.11).

3 Particle filters

From this point on, we adopt the convention that whenever the index j is used, we mean "for all $j = 1, \dots, M$". We present SMC methods designed to approximate the sequence of filtered distributions $\pi^\theta(x_n|y_{0:n})$: at the end of each time step n, the particle filters discussed hereafter return M particles X_n^j with weights w_n^j that can be used to approximate for instance

- the filtered distribution $\pi^\theta(x_n|y_{0:n})$ by the finite mixture of weighted Dirac masses

$$\widehat{\pi}(\mathrm{d}x_n|y_{0:n}) = \sum_{j=1}^{M} w_n^j \delta(X_n^j, \mathrm{d}x_n),$$

- integrals such as $I_n = \int h(x_n)\pi(x_n|y_{0:n})\,\mathrm{d}x_n$, with $h \in L^1(\pi(\cdot|y_{0:n}))$, by

$$\widehat{I}_{n,M} = \sum_{j=1}^{M} w_n^j h(X_n^j).$$

3.1 Sequential Importance Sampling (SIS)

Conception

The SIS filter (sometimes also called Bayesian Importance Sampling) is a direct application of the calculations shown in the previous section: it relies solely upon the sequential use of the self-normalised importance sampling technique. The details are given in Algorithm 3.1.

Algorithm 3.1 (Sequential Importance Sampling (SIS) for filtering)**.**

At time $n = 0$

1. Sample $X_0^j \sim q^\theta(x_0|y_0)$.
2. Compute $\widetilde{w}_0^j = \dfrac{g_0^\theta(y_0|X_0^j)\mu^\theta(X_0^j)}{q^\theta(X_0^j|y_0)}$ and set $w_0^j \leftarrow \dfrac{\widetilde{w}_0^j}{\sum_{k=1}^M \widetilde{w}_0^k}$.

At time $n \geq 1$

1. Sample $X_n^j \sim q^\theta(x_n|x_{0:n-1}, y_{0:n})$.
2. Compute $\widetilde{w}_n^j = w_{n-1}^j \dfrac{g_n^\theta(y_n|X_n^j) f_n^\theta(X_n^j|X_{n-1}^j)}{q^\theta(X_n^j|X_{0:n-1}^j, y_{0:n})}$ and set $w_n^j \leftarrow \dfrac{\widetilde{w}_n^j}{\sum_{k=1}^M \widetilde{w}_n^k}$.

At each time step, new particles are first simulated conditionally to the old ones to represent the predictive distribution of the upcoming state and, as the observation becomes available, their weights then get readjusted to represent the filtered distribution.

Prediction

The estimation of the predicted distribution $\pi^\theta(x_{n+\tau}|y_{0:n})$ ($\tau \geq 1$) can also be computed from the estimation of the filtered distribution up to time n. The principle, described for instance in Doucet (1998), is identical in essence to that developed in Durbin and Koopman (2001) for Kalman filters. Since the observations at times $n+1, \ldots, n+\tau$ are not yet available, no correction may take place after the predictions of the state that involve the transition densities $f_{n+\tau}^\theta, \ldots, f_{n+1}^\theta$: formally, the terms $g_{n+\tau}^\theta, \ldots, g_{n+1}^\theta$ vanish. The details are given in Algorithm 3.2. Observe that in this case, the importance density $q^\theta(x_{n+\tau}|x_{0:n+\tau-1}, y_{0:n})$ needs to be chosen so as not to involve the yet unknown values of the upcoming observations $y_{n+1:n+\tau}$.

Algorithm 3.2 (Sequential Importance Sampling (SIS) for predicting)**.**

At time $n \geq 0$**, for** $\tau = 1, \ldots$

1. Sample $X_{n+\tau}^j \sim q^\theta(x_{n+\tau}|x_{0:n+\tau-1}, y_{0:n})$.
2. Compute $\widetilde{w}_{n+\tau}^j = w_{n+\tau-1}^j \dfrac{f_{n+\tau}^\theta(X_{n+\tau}^j|X_{n+\tau-1}^j)}{q^\theta(X_{n+\tau}^j|X_{0:n+\tau-1}^j, y_{0:n})}$ and set $w_{n+\tau}^j \leftarrow \dfrac{\widetilde{w}_{n+\tau}^j}{\sum_{k=1}^M \widetilde{w}_{n+\tau}^k}$.

Missing observations

When dealing with a missing observation, the SIS filter requires little modification: when observation Y_n is missing, the corresponding state X_n is predicted using Algorithm 3.2 since $\pi^\theta(x_n|y_{0:n-1})$ is the only accessible density under such circumstances. This leads to Algorithm 3.3.

Algorithm 3.3 (Sequential Importance Sampling (SIS) for filtering with missing observations)**.**

At time $n \geq 0$, if observation Y_n is missing

1. Sample $X_n^j \sim q^\theta(x_n | x_{0:n-1}, y_{0:n-1})$.
2. Compute $\widetilde{w}_n^j = w_{n-1}^j \dfrac{f_n^\theta(X_n^j | X_{n-1}^j)}{q^\theta(X_n^j | X_{0:n-1}^j, y_{0:n-1})}$ and set $w_n^j \leftarrow \dfrac{\widetilde{w}_n^j}{\sum_{k=1}^M \widetilde{w}_n^k}$.

Comments

The major drawback of the SIS filter comes from the fact that the distribution of the weights degenerates, with the variance of the importance weights increasing over time (see Doucet et al., 2000) meaning that the estimated distributions become less and less unreliable: after a few iterations, all but one of the normalised importance weights are close to zero. An important fraction of the calculations involved in the algorithm is thus dedicated to particles whose contributions to the estimation are almost null, making the SIS particle filter an impractical estimation procedure at best.

3.2 Monitoring the degeneracy

To alleviate the degeneracy problem that we outlined, additional steps are traditionally implemented into Algorithm 3.1. Since adding these new steps comes at a non negligible computational cost, it is important to somehow monitor how badly the weight distribution degenerates at a given time step, because it is usually interesting to ignore the degeneracy problem unless it reaches a given threshold.

A popular rule of thumb, first introduced in Kong et al. (1994) and later copiously reprised in the literature (see for instance Doucet et al., 2000; Chen, 2003; Liu, 2008), is to consider the so-called effective sample size based on the normalised weights w_n^j at time step n and defined by

$$\frac{M}{1 + \mathrm{Var}^{q^\theta(\cdot | y_{0:n})}[w_n^1]}.$$

This quantity is usually numerically approximated by the following estimate

$$\mathrm{ESS}(n) = \frac{1}{\sum_{k=1}^M (w_n^k)^2}. \tag{3.1}$$

It ranges from M (reached when all the particles share equal weights of value 1) to $1/M$ (reached when a single particle is given the whole probability mass of the sample, with a weight of 1).

A related degeneracy measure is the coefficient of variation (found in Kong et al., 1994; Liu and Chen, 1995), ranging from 0 to $\sqrt{M-1}$, that is given by

$$\mathrm{CV}(n) = \sqrt{\frac{1}{M} \sum_{k=1}^M (M w_n^k - 1)^2}, \tag{3.2}$$

and satisfies to

$$\mathrm{ESS}(n) = \frac{M}{1 + \mathrm{CV}(n)^2}. \tag{3.3}$$

The Shannon entropy of the importance weights, ranging from $\log M$ to 0, is sometimes also mentioned. It is defined by

$$\mathcal{E}(n) = -\sum_{k=1}^{M} w_n^k \log w_n^k. \tag{3.4}$$

Cornebise (2009) recently proved that the criteria (3.2) and (3.4) are estimators of the χ^2-divergence and the Kullback-Leibler divergence between two distributions which are associated with the importance and target densities of the particle filter.

The evaluation of one (or more) of these criteria is introduced at each time step, with the additional procedures that we discuss next taking place if and only if the criterion reaches a certain fixed threshold so as to reduce the additional computational burden. The most common threshold found in the literature is $\text{ESS}(n) < 0.5M$. Examples illustrating the behaviours of these criteria are given later in Figures 3, 4 and 5.

3.3 Resample step

A resampling step is most often introduced into Algorithm 3.2 to help and fight the degeneracy problem. The aim of this resampling step is to favour the living of the interesting particles (the ones with more important weights, that are more representative of the targeted distribution) and encourage the dying of the not so interesting particles so as to focus the computational effort upon particles that matter most for the estimation. The resampling method has to be carefully chosen, in particular it should not introduce any bias in the final estimate as mentioned in Doucet et al. (2000)

During this new step, particles are resampled according to their weights: a particle with an important weight is more likely to appear (and "survive") in the new sample generated, possibly more than once, whereas a particle the weight of which is close to zero is more likely not to be drawn at all (and "die") from a given time step to the next.

Chen (2003) mentions that there are a few resampling schemes available in the literature. It is important to note that even though resampling might alleviate the degeneracy problem, it also brings extra random variation to the samples of particles. As a consequence, the filtered quantities of interest should preferably be computed before resampling and not after. We only present the details of the multinomial and residual resampling schemes.

Multinomial resampling

Multinomial resampling is the most popular resampling scheme, most likely because it is the easiest to both understand and implement: at a given time step, it suffices to simulate a discrete random variable which takes values X_n^k with probability w_n^k. The details of multinomial resampling are given in Algorithm 3.4 where only the new step is described.

Algorithm 3.4 (Multinomial resampling step)**.**

At time $n \geq 0$

3. Sample $Z_n^j \sim \sum_{k=1}^{M} w_n^k \delta(X_n^k, \mathrm{d}x)$.

   Replace $X_n^j \leftarrow Z_n^j$ and $w_n^j \leftarrow 1/M$.

Used as is, it leads to the well-known Sampling Importance Resampling (SIR) filter, sometimes also called Bootstrap filter, that can be found in Gordon et al. (1993). A straightforward implementation of the multinomial resampling has complexity $\mathrm{O}(M \log M)$: it is indeed equivalent to simulating M draws from a discrete random variable Z_n such that $\mathbb{P}(Z_n = k) = w_n^k$.

A trivial implementation for such simulations requires first to draw $U_n^1, \ldots, U_n^M$ i.i.d. with uniform distribution and then to find the indexes i_n^j for which $U_n^j \in]\sum_{k=1}^{i-1} w_n^k, \sum_{k=1}^{i} w_n^k]$. Finding the indexes i_n^j has only complexity $\mathrm{O}(M)$ when the random variables are U_n^j are ordered, but ordering these random variables has complexity $\mathrm{O}(M \log M)$ at least, using for instance the quicksort algorithm (see Hoare, 1962).

A practical implementation of the multinomial resampling is proposed in Doucet (1998) which circumvents the naive need of sorting M i.i.d. random variables with uniform distribution and relies upon a direct simulation trick instead. The complexity of the SIR filter can hence be reduced from $\mathrm{O}(M \log M)$ (naive implementation using quicksort) to only $\mathrm{O}(M)$ which saves a significant amount of computational resources.

Residual-multinomial resampling

Residual-multinomial resampling is proposed in Liu and Chen (1998) to reduce the extra variance introduced by the resamping step. It is partially deterministic as opposed to the multinomial resampling and is formulated below. Let $\lfloor x \rfloor$ designate the integer part of a real number x and define for any $n \geq 0$:

$$R_n = \sum_{k=1}^{M} \lfloor M \cdot w_n^k \rfloor, \qquad \overline{w}_n^j = \frac{M \cdot w_n^j - \lfloor M \cdot w_n^j \rfloor}{M - R_n}.$$

Algorithm 3.5 (Residual-multinomial resampling step)**.**

At time $n \geq 0$

3. Copy $\lfloor M \cdot \widehat{w}_n^j \rfloor$ particles $\widehat{X}_n^j$. (R_n particles are thus allocated, say $Z_n^1, \ldots, Z_n^{R_n}$).

   Sample the remaining particles $Z_n^{R_n+1}, \ldots, Z_n^M \sim \sum_{k=1}^{M} \overline{w}_n^k \delta(X_n^k, \mathrm{d}x)$.

   Replace $X_n^j \leftarrow Z_n^j$ and $w_n^j \leftarrow 1/M$.

The details of residual-multinomial resampling are given in Algorithm 3.5 where only the new step is described. In essence, particles with weights greater than $1/M$ are forced into the new sample, and the rest is allocated at random, depending on the remaining probability mass available. Note that the last part of a residual resampling step is basically a multinomial resampling step on the residual probability mass, hence the name.

It is shown to be computationally cheaper than the multinomial resampling, due to the fact that only a fraction of the M particles are randomly allocated. It does not introduce any bias for the estimation and has the added advantage of having a lower variance than that of the multinomial resampling (see Douc and Cappe, 2005, for the proofs).

Other resampling techniques

Stratified and systematic resampling also offer an alternative to the multinomial resampling scheme (see Kitagawa (1996) and Carpenter et al. (1999) or Chen (2003) for a more general

overviews). Systematic resampling appears to be another popular choice in the literature for computational reasons even though its variance is not guaranteed to be smaller than that of the multinomial resampling as stated in Douc and Cappe (2005). A short study of these techniques and a numerical comparison of their performance on an example are offered in Cornebise (2009). Note that residual versions of these techniques also exist, where they are substituted to the multinomial sampling used in the second half of Algorithm 3.5.

Limitations of the resampling procedure

The resampling procedure alleviates the degeneracy problem but also introduces practical and theoretical issues (as mentioned in Doucet et al., 2000, for example). From a practical point of view, resampling very obviously limits the opportunity of parallelisation of the algorithm. From a theoretical point of view, simple convergence results are lost due to the fact that after one resampling step the particles are not independent anymore. Moreover, resampling causes the particles with high importance weights to be statistically selected many times: the algorithm thus suffers from the so-called loss of diversity.

3.4 Move step

The loss of diversity among the particles following the resample step is usually addressed in the literature with the introduction of yet another additional move step into the algorithm: the idea behind it is to rejuvenate the diversity after the particles have been resampled.

Using MCMC

Gilks and Berzuini (2001); Doucet et al. (2001) present the so-called Resample-Move algorithm in which an MCMC step is used after resampling. This new step relies upon the use of Markov transition kernels with appropriate invariant distributions. Moving the particles according to such kernels formally guarantees the particles still target the distribution of interest but also give them an additional chance to move towards an interesting region of the state space while increasing the diversity of the sample at the cost of an increased computational burden. Doucet and Johansen (2011) underline the possibility of using even non ergodic MCMC kernels for this purpose and also propose to go a step further and rejuvenate not only the current state but also some of the (immediate) past states with the so-called Block Sampling (the computational cost of which is thus even greater).

Using regularisation

Another approach to deal with the loss of diversity is based upon regularisation techniques. Let us define for $x, x^* \in \mathcal{X} \subset \mathbb{R}^{n_x}$

$$K_h(x, x^*) = h^{-n_x} \cdot (\det \Sigma_n)^{-1/2} \cdot K\left(\Sigma_n^{-1/2} \cdot \frac{x - x^*}{h}\right)$$

where K is usually a smooth symmetric unimodal positive kernel of unit mass (hence a probability measure), h is the bandwidth of the kernel, and Σ_n designates the empirical covariance matrix of the sample (see Silverman, 1986, for the idea of whitening the sample via Σ_n).

Algorithm 3.6 (Regularisation step)**.**

At time $n \geq 0$

4. Sample $\epsilon_n^j \sim K(x)$, and set $Z_n^j \leftarrow X_n^j + h \cdot \Sigma_n^{1/2} \cdot \epsilon_n^j$.
   Replace $X_n^j \leftarrow Z_n^j$ and keep $w_n^j \leftarrow w_n^j$.

Gordon et al. (1993) originally referred to that step as "jittering" since it adds a small amount of noise to each resampled particle. Note that, when used together with the multinomial resampling scheme described in Algorithm 3.4, the resulting combination of the two steps can be reformulated as described in Algorithm 3.7: it is then equivalent to resampling new particles from the smoothed estimated target distribution (using kernel density K).

Algorithm 3.7 (Alternate formulation for the combination of Algorithms 3.4 and 3.6)**.**

At time $n \geq 0$

3+4. Sample $Z_n^j \sim \sum_{k=1}^{M} w_n^k K_h(X_n^k, x)$.
   Replace $X_n^j \leftarrow Z_n^j$ and $w_n^j \leftarrow 1/M$.

The choice of both the kernel smoothing density K and the bandwidth h obviously has a big impact on the algorithm. The idea is to resample from a density estimated from the particles at time step n that best approximates the true target density. Picking $K(\cdot) = \delta(\cdot, 0)$ the Dirac mass at the origin turns the regularised SMC filter back into a simple SMC filter. From a general point of view, we would like the estimated density to converge as fast as possible towards the true target density as M goes to $+\infty$, since the number of particles will necessarily be limited by the computational resources.

For the Gaussian kernel (among others), Silverman (1986) shows it is possible to compute the optimal bandwidth to use, i.e. the bandwidth that minimises the variance of the density estimate. Although it could be argued that selecting a proper bandwidth is a difficult task, this optimal bandwidth yields good results in practise and at least provides a rough idea about the scaling of h. As is the case with kernel density estimates, the choice of h directly influences the trade-off made between variance and bias of the estimate: if h is chosen too small, the loss of diversity will still be severe, and if h is chosen too large, the filtered density will roughly be estimated as a single kernel, hence introducing a severe bias into the estimation.

The use of the Epanechnikov kernel, proportional to $1 - \|x\|^2$ on the unit ball of the state space, is recommended in Silverman (1986) because it is asymptotically the most efficient, and Doucet (1998) claims it can be difficult to choose a "good" kernel. However, we advocate the use of the Gaussian kernel whenever possible for computational reasons: simulations from the Gaussian kernel are readily available on most machines and come at a computationally cheaper price than simulations from the Epanechnikov kernel. But the non optimality of Gaussian kernel does not outbalance its ease of use, since the choice of the kernel neither affects the order of the bandwidth nor the rate of convergence as stated in DasGupta (2008).

From a general point of view it is also possible to choose a n_x-dimensional kernel under the form of a product of n_x 1-dimensional (possibly distinct) kernels. Such a choice is preferable when some coordinates of the state are bounded. It allows for easier simulations on these coordinates using dedicated truncated kernels whereas a straightforward accept-reject algorithm could turn out to be highly inefficient (with a low acceptance rate) depending on the boundaries of the state space.

Finally, the regularisation can also be done before resampling thus resulting in the so-called pre-regularised particle filter (pre-PRF) as opposed to the post-regularised particle filter presented here. Theoretical convergence results about these regularised filters are available in Oudjane (2000) and Rossi (2004)

3.5 Detection and removal of outliers

In order to deal with the sensitivity of the particle filters to outliers, we propose a new additional rule at the end of step 2 of Algorithm 3.1. Its role is to make sure that outliers do not lead to a fully degenerated situation, that the algorithm would not recover from. The details of it are given in Algorithm 3.8 where only the additional rule is described.

Algorithm 3.8 (Online detection and removal of outliers.)**.**

At time $n \geq 0$

If the degeneracy problem is critical, consider the observation y_n as missing (see Algorithm 3.3) and rewind back to step 1.

The rule applies only to situations where the degeneracy of the sample is critical: when the importance density chosen is the prior density, it triggers only when the current observation is not predicted efficiently. In that case, we proceed as if the observation was missing. In practise the degeneracy problem is deemed critical when a criterion such as $\mathrm{ESS}(n) < \epsilon \cdot M$ is met, with $\epsilon > 0$ very small.

Observations that do not agree with the model are thus detected online and ignored to prevent immediate degeneracy. This trick is in a way similar to the one introduced in Hu et al. (2008, 2011) where a resample step is iterated until the likelihood of the current observation with regard to the resampled particles is above a given threshold. While both techniques ensure that the particles do not collapse when an outlier is met, the cost paid is different for each. The alteration proposed in Hu et al. (2008, 2011) can be computationally expensive (with an unbounded runtime) but the observation ends up being taken into account, while our own modification is definitively cheaper (with a guaranteed fixed runtime) but discards the observation at hand when it strongly disagrees with the current state of the model. A significant change of state will still be detected in the long run, because considering the observation y_n as missing automatically implies the variance of the state grows larger (which means that, if it were to be repeated, the outlying observation, would seem more likely at the next time step, with regard to the new state).

3.6 Choice of the importance density

As previously stated the particle filters rely on the introduction of an importance density that was chosen of the form given in (2.10) i.e.

$$q^\theta(x_{0:n}|y_{0:n}) = q^\theta(x_0|y_0) \prod_{k=1}^{n} q^\theta(x_k|y_{0:k-1}, y_{0:k}).$$

Choosing carefully the importance density q^θ can help reduce the variance of the importance weights and thus alleviate the degeneracy problem. As the choice is abundantly discussed in the literature, we only selected three representative alternative among the many that are available.

Prior density

A default choice consists of taking $q^\theta(x_0|y_0) = \mu^\theta(x_0)$ and $q^\theta(x_n|x_{0:n-1}, y_{0:n}) = f_n^\theta(x_n|x_{n-1})$, i.e. taking the prior density (1.1) of the model as the importance function. This choice works even with missing data (as it does not depend on y_n) and leads to much simpler calculations for the update of the importance weights as can be seen directly in the formulae given in Algorithm 3.9.

Algorithm 3.9 (Sequential Importance Sampling (SIS) for filtering, using the prior density as the importance density)**.**

At time $n = 0$

1. Sample $X_0^j \sim \mu^\theta(x_0)$.
2. Compute $\widetilde{w}_0^j = g_0^\theta(y_0|X_0^j)$ and set $w_0^j \leftarrow \dfrac{\widetilde{w}_0^j}{\sum_{k=1}^M \widetilde{w}_0^k}$.

At time $n \geq 1$

1. Sample $X_n^j \sim f_n^\theta(x_n|X_{n-1}^j)$.
2. Compute $\widetilde{w}_n^j = w_{n-1}^j g_n^\theta(y_n|X_n^j)$ and set $w_n^j \leftarrow \dfrac{\widetilde{w}_n^j}{\sum_{k=1}^M \widetilde{w}_n^k}$.

Note that using the prior density as the importance density makes the algorithm propose new particles in a blind way: the new particles are simulated around the current state, not around the upcoming targeted state. With such a choice of importance density, the algorithm becomes especially sensitive to outliers. An annealed version of the prior distribution is proposed in Chen (2003) to help deal with some situations where prior and likelihood do not agree.

Optimal density

Although popular, the choice of the prior density is not optimal: the optimal choice is given by $q^\theta(x_0|y_0) = \pi^\theta(x_1|y_1)$ and $q^\theta(x_n|x_{0:n-1}, y_{0:n}) = \pi^\theta(x_n|y_n, x_{n-1})$ in the sense that it minimises the variance of the importance weights conditional upon the past states and the past observations as can be seen in Doucet et al. (2000). The idea underlying this choice is to take into account the upcoming observation so that particles are not blind to the upcoming state anymore. Most of the time sampling from these optimal distributions is not an option however, and it is usually recommended to approximate them if possible. For example Pitt and Shephard (1999) propose the so-called Auxiliary Particle Filter (APF) which essentially reverses the sampling and resampling phase mentioned in the previous algorithms (see Whiteley and Johansen, 2011). It relies upon the introduction of an augmented state that is used to select the most representative particles in the sense that their predictive likelihoods are large. Doucet et al. (2000) use the Extended Kalman filter to derive a Gaussian approximation (relying on a local linearisation of the state space model) and van der Merwe et al. (2001) discuss the use of the Unscented Kalman Filter to obtain such approximations (see Wan and van der Merwe, 2000, for the details about implementing the UKF).

Independent density

Let us mention that it is also possible to use an independent importance density (independent with regard to the states and observations) but it is strongly recommended to avoid such a choice because it "ignores" both the current and the upcoming states (see Doucet et al., 2000)

3.7 Parameter estimation

Thus far, state estimation was discussed conditionally to the fact that the parameter θ was known. However, θ is often unknown and has to be estimated together with the state of the dynamic model. Kantas et al. (2009) offers a comparative review of the possible choices available for parameter estimation, presenting maximum likelihood and Bayesian parameter estimation in the context of an offline or online procedure. We provide here only a brief overview of the Bayesian parameter estimation and direct the interested reader to the original paper for the complete discussion.

One of the first approach considered in the literature for parameter estimation is to extend the state X_n at time n into a new state $Z_n = (X_n, \theta_n)$ with initial distribution $\mu^{\theta_0}(x_0)\pi(\theta_0)$ and transition density $f^{\theta_n}(x_n|x_{n-1}) \cdot \delta(\theta_n, \theta_{n-1})$ and then estimate this new extended model with a standard SMC filter as in Kitagawa (1996). Even though the approach is theoretically sound as claimed in Rossi (2004); Kantas et al. (2009), it can lead to a strong loss of diversity problem on the coordinate θ when no move step is implemented as the parameter space is only explored at the initialisation of the algorithm, making such an approach often unusable.

The addition of a move step into the algorithm provides a satisfying solution to this problem as can be seen in Rossi (2004) who successfully applied the kernel regularisation technique, or in Andrieu et al. (1999) who makes use of MCMC techniques in a move step to update the parameter value. The regularisation can also be combined with a judicious choice of importance density such as with the APF (see Pitt and Shephard, 1999) to provide remarkably accurate parameter estimation (as shown in Casarin and Marin, 2009; Whiteley and Johansen, 2011, for example). Another option is to force a fictitious small dynamic upon the parameter as described in Kitagawa (1998); Higuchi (2001) so that it is artificially allowed to evolve over time, even though Kantas et al. (2009) rightly remarks that modifying the model in such a way makes it hard to quantify how much bias is introduced in the resulting estimates.

A more recent way of estimating the parameter together with the state relies upon the use of so-called Particle Markov Chain Monte Carlo (PMCMC) methods found in Andrieu et al. (2010). These methods are computationally expensive both in term of storage and calculations, because their computational cost typically grows with time as underlined in Chopin et al. (2012), and thus are less fit for online estimation than some standard SMC filter: the most basic PMCMC method, known as the Particle Marginal Metropolis-Hastings (PMMH) sampler and described in Kantas et al. (2009), involves running an SMC filter for each step of a Metropolis-Hastings algorithm used to propose a new value of the parameter θ.

3.8 Summary

In the end, keeping in mind that the original aim is the online estimation and prediction, we implemented an algorithm not too computationally expensive. We chose the importance density to be the prior density of the model and included a residual resample step coupled with a regularisation move step, that triggered whenever $\text{ESS}(n) < 0.5M$ unless $\text{ESS}(n) < 0.001M$, in which case the current observation was instead considered an outlier and thus treated as missing. For the regularisation step (see Algorithm 3.6) we use a Gaussian kernel K with a bandwidth h optimally chosen for the mean integrated squared error (see Silverman, 1986, Chapter 4). As for the parameter estimation problem, we opted for the solution of extending the state-space and introduced no artificial dynamic on the parameter θ (thus using $\theta_n = \theta_{n-1}$). In practise, this results in the disappearance of the θ superscript on densities μ, f_n and g_n in the description of Algorithm 3.10. We did however test the introduction of an artificial dynamic on the parameters but observed no changes in the measured overall performance. Note that the regularisation step mentioned above applies to the extended state (i.e. including the parameter).

Algorithm 3.10 (Particle filter used for our application)**.**

At time $n = 0$

1. Sample $\widehat{X}_0^j \sim \mu(x_0)$.
2. Compute $\widetilde{w}_0^j = g_0(y_0|X_0^j)$ and set $\widehat{w}_0^j \leftarrow \dfrac{\widetilde{w}_0^j}{\sum_{k=1}^M \widetilde{w}_0^k}$.
   - if $\widehat{\mathrm{ESS}}(0) < 0.001M$, set $X_0^j \leftarrow \widehat{X}_0^j$ and $w_0^j \leftarrow 1/M$.
   - if $0.001M \leq \widehat{\mathrm{ESS}}(0) < 0.5M$, use residual-multinomial resample (see Algorithm 3.5) and regularisation move (see Algorithm 3.6) steps to set X_0^j and w_0^j.
   - if $0.5M \leq \widehat{\mathrm{ESS}}(0)$, set $X_0^j \leftarrow \widehat{X}_0^j$ and $w_0^j \leftarrow \widehat{w}_0^j$.

At time $n \geq 1$

1. Sample $\widehat{X}_n^j \sim f_n(x_n|X_{n-1}^j)$.
2. Compute $\widetilde{w}_n^j = w_{n-1}^j g_n(y_n|X_n^j)$ and set $\widehat{w}_n^j \leftarrow \dfrac{\widetilde{w}_n^j}{\sum_{k=1}^M \widetilde{w}_n^k}$.
   - if $\widehat{\mathrm{ESS}}(n) < 0.001M$, set $X_n^j \leftarrow \widehat{X}_n^j$ and $w_n^j \leftarrow w_{n-1}^j$.
   - if $0.001M \leq \widehat{\mathrm{ESS}}(n) < 0.5M$, use residual-multinomial resample (see Algorithm 3.5) and regularisation move (see Algorithm 3.6) steps to set X_n^j and w_n^j.
   - if $0.5M \leq \widehat{\mathrm{ESS}}(n)$, set $X_n^j \leftarrow \widehat{X}_n^j$ and $w_n^j \leftarrow \widehat{w}_n^j$.

4 Application

In this Section we describe an application of particle filters for electricity load forecasting. We quickly describe the data used for our experimentation and the two similar models that were estimated using Algorithm 3.10, deal with the problem of initialising the particle filter and discuss the results obtained.

4.1 Data

Time range

The data chosen for the application contain the consolidated half-hourly electricity load at the "EDF" perimeter over the period ranging from 04/01/2006 to 03/31/2011 which represents five years worth of measurements, with 48 points per day. Note that only an estimation of the load is available in real time. The consolidated data correspond to the true (not estimated) signal that is available only three weeks later.

Daytypes

The calendar used for the application provides nine distinct daytypes, the list of which is given in Table 1. In essence, this is a very basic calendar that models a single bank-holidays effect where more detailed calendars would model multiple different ones. Although such a basic calendar

arguably does not reflect the whole variety of daytypes, it is detailed enough for our purpose and helps keep the dimension of the model we propose as low as possible.

#	day
0	mon.
1	tue.-wed.-thu.
2	fri.

#	day
3	sat.
4	sun.
5	before BH

#	day
6	BH
7	after BH
8	between BH and a weekend

Table 1: Daytypes provided by the basic calendar used in the application. BH stands for a bank holiday.

Note that the operational model used by EDF also require the precise specification of daytypes and so-called offsets, the latter being used to model breakpoints (see Bruhns et al., 2005, for the details).

From here on, we will call bank-holidays, the instants in the calendar where specific information is needed for the operational model to be correctly estimated and predicted. These instants essentially correspond to bank-holidays (daytypes from 5 to 8, hence the name), or the summer and winter holiday breaks and are signalled on Figure 2.

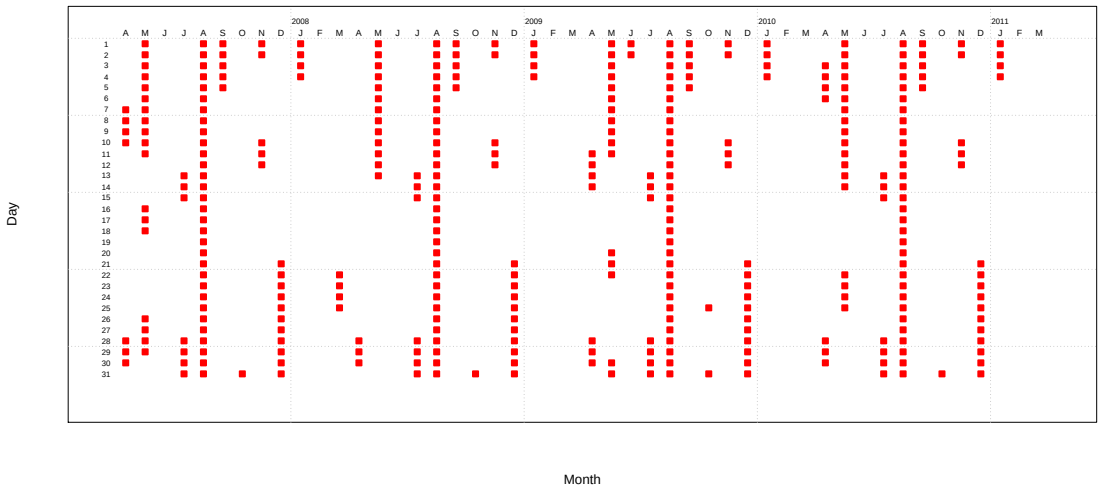

Figure 2: Repartition of the bank holidays amidst the calendar from 04/01/2007 to 03/31/2011. Each column represents a month.

4.2 Dynamic model

The formulation of the model that we consider was inspired by the works of Bruhns et al. (2005); Dordonnat (2009); Launay et al. (2012b). It features three parts (seasonality, heating, cooling) similarly to what was done in Launay et al. (2012b) and includes a two layers dynamic on the two most relevant parts with regard to the French electricity load for each of the 48 half-hours (or instants) within a day. The 48 corresponding independent model is estimated and predicted in parallel, using the calendar and temperature information described above, the results being aggregated back together at the end of the process. The dimensions of the parameter and state

spaces were voluntarily kept small: the goal is ultimately to provide competitive one-day-ahead predictions for the electricity load based on a model as parsimonious as possible within a rather general framework.

We denote $\mathcal{N}(\mu, \Sigma)$ the Gaussian distribution with mean μ and variance Σ, and $\mathcal{N}(\mu, \Sigma, \mathcal{S})$ the corresponding truncated Gaussian distribution the support of which is $\mathcal{S}$. For each half-hour, and removing the now superfluous i subscript, the model that we consider is defined by

$$y_n = x_n + \nu_n, \tag{4.1}$$

where $\nu_n \sim \mathcal{N}(0, \sigma^2)$ and where the state x_n is made of three parts

$$x_n = x_n^{\text{season}} + x_n^{\text{heat}} + x_n^{\text{cool}},$$

that are defined by

$$\begin{aligned} x_n^{\text{season}} &= s_n \cdot \kappa_{\text{daytype}_n} \\ x_n^{\text{heat}} &= g_n^{\text{heat}} (T_n^{\text{heat}} - u^{\text{heat}}) \mathbb{1}_{]T_n^{\text{heat}}, +\infty[}(u^{\text{heat}}) \\ x_n^{\text{cool}} &= g^{\text{cool}} \Delta_n^{\text{cool}}. \end{aligned}$$

The various components obey the following prior dynamic

$$\begin{aligned} s_n &= s_{n-1} + \epsilon_n^s, & \epsilon_n^s &\sim \mathcal{N}\left(0, \sigma_{s,n}^2,] - s_{n-1}, +\infty[\right) \\ g_n^{\text{heat}} &= g_{n-1}^{\text{heat}} + \epsilon_n^g, & \epsilon_n^g &\sim \mathcal{N}\left(0, \sigma_{g,n}^2,] - \infty, -g_{n-1}^{\text{heat}}[\right) \\ \sigma_{s,n} &= \sigma_{s,n-1} + \eta_n^s, & \eta_n^s &\sim \mathcal{N}\left(0, \sigma_s^2,] - \sigma_{s,n-1}, +\infty[\right) \\ \sigma_{g,n} &= \sigma_{g,n-1} + \eta_n^g, & \eta_n^g &\sim \mathcal{N}\left(0, \sigma_g^2,] - \sigma_{g,n-1}, +\infty[\right) \end{aligned}$$

where daytype_n, T_n^{heat} and Δ_n^{cool} correspond to the exogenous variables that we already discussed:

- denoting N_{daytype} the number of different daytypes featured in the calendar provided, $\text{daytype}_n \in \mathbb{N}$ takes a finite number of values between 0 and $N_{\text{daytype}} - 1$ and represents the class to which the day n belongs with regard to the calendar ;
- $T_n^{\text{heat}} \in \mathbb{R}$ is the temperature used to compute the heating part of the model, which is precomputed at EDF as a mixture of exponentially smoothed signals ;
- $\Delta_n^{\text{cool}} \in \mathbb{R}_+$ provides the cooling degrees needed to compute the cooling part of the model.

Using the definitions and notations introduced in Section 1, the parameter of the model is given by $\theta = (\sigma_s, \sigma_g, g^{\text{cool}}, u^{\text{heat}}, \kappa, \sigma)$, these quantities are assumed constant over time in the model. At time n, the state of the model is given by x_n whose components $(s_n, g_n, \sigma_{s,n}, \sigma_{g,n}) \in \mathbb{R}^4$ are the quantities that vary over time according to the dynamic specified. All these quantities are unknown and are to be estimated.

The model (4.1) includes a seasonal part x_n^{season} that is essentially made of a signal s_n, the dynamic prior of which is a random-walk process whose standard deviation $\sigma_{s,n}$ itself evolves as a random-walk. s_n is multiplied by a coefficient $\kappa_{\text{daytype}_n}$ that depends on the daytype of the current observation to model the difference in behaviour between the electricity load on weekdays and weekends or holidays. For identifiability reason, the sum of the coefficients κ_j is fixed so that

$$\frac{1}{N_{\text{daytype}}} \sum_{j=1}^{N_{\text{daytype}}} \kappa_j = 1.$$

Note that s_n essentially replaces the truncated Fourier series featured in Launay et al. (2012b).

The model (4.1) also includes two weather-related parts to account for the influence of low (heating part) and high temperatures (cooling part) upon the electricity load : the heating part x_n^{heat} is based upon a truncated difference between the temperature T_n^{heat} and a heating threshold u^{heat}, as studied in Launay et al. (2012a). This difference is multiplied by a gradient g_n^{heat} whose dynamic is similar to that of s_n: the prior is a random-walk whose standard deviation $\sigma_{g,n}$ itself evolves as a random-walk. Because the cooling effect in France is of a lesser magnitude than the heating effect, the corresponding model for the cooling part x_n^{cool} is simpler: the precomputed truncated difference Δ_n^{cool} is given to the model and multiplied by a cooling gradient g^{cool}.

Notice that to ensure the different quantities involved kept consistent signs throughout time, we specifically used truncated Gaussian distributions. In particular, this means that the random-walks featured in the dynamic are not symmetric and hence that the mean of the state is a priori expected to slightly evolve over time. The constraint on ϵ_n^s and ϵ_n^g can of course easily be lifted if need be, and does not affect the overall predictive performance of the model in any way.

4.3 Initialisation of the particle filter

As was already discussed, the degeneracy of the particles sample over time is a serious matter. The choice of the initial distribution of the state is thus of the utmost importance because a strong disagreement between this distribution and the first filtered distribution could lead to sample degeneracy after only a single time step. Two solutions are theoretically viable to choose the initial prior distribution: one may choose either a vague or an informative distribution.

1. on one hand, a vague prior has the advantage of not biasing the dynamic model before the first observations. However, the variance of the initial distribution of the state being very large, the sequence of posterior variances of the filtered distributions of the state tend to decrease very quickly at first. From an SMC filter point of view, one has to use a very large sample of particles to cover at the same time the regions of the state space with prior highest probability and with posterior highest probability: a vague initialisation thus requires the use of a massive number of particles.

2. on the other hand, designing an informative distribution is a totally different task, but still not a trivial one: one has to keep in mind that a "bad" choice of initial distribution may lead to immediate degeneracy. Intuitively, the ideal solution would be to dispose at time $n = -1$ of a filtered distribution $\pi^\theta(x_{-1}|y_{-1}, \ldots, y_{-m})$ to use it as a the initial distribution at time $n = 0$. Such a choice is of course not possible because observations are only available for time $n = 0, \ldots, N$.

Note that the trick of ignoring outliers introduced into the particle filter (see Algorithm 3.8) does not alleviate the problem of initialisation, since it can only increase the variance if it is used.

We thus opted for a more general procedure that allows for an automated initialisation of the particles sample to a fitting state space region from time $n = n_0$, and that combines the two approaches mentioned above to retrieve the benefits of both:

1. we use a vague distribution to estimate the smoothed distribution up to time $n = n_0 - 1$ using open-source MCMC generic software such as BUGS (Lunn et al., 2000) or JAGS (Plummer, 2003): we typically chose $n_0 = 365$ so that the variance of the filtered distribution at time $n = n_0 - 1$ is already small enough not to require the use of a massive amount of particles ;

2. after this first MCMC initialisation phase, we retrieve particles (approximately) distributed along the filtered distribution of the state at time $n = n_0 - 1$: this distribution is the one used (through these particles) to initialise the SMC filter at time n_0.

There is however a price to pay for solving the initialisation problem in such a way. First we have to use MCMC to initialise the particle filter and second it makes it hard to use the particle filter on a time series with few observations. Note that the first issue raised is but rhetorical: MCMC, even if expensive, has to be run only once, and not at each time step.

Initial distribution for the MCMC estimation

The initial distribution envisioned for the dynamic model (4.1) is vague and specified by:

$$\begin{aligned} s_0, g^{\text{cool}} &\sim \mathcal{N}(0, 10^8, \mathbb{R}_+) \\ g_0^{\text{heat}} &\sim \mathcal{N}(0, 10^8, \mathbb{R}_-) \\ u^{\text{heat}} &\sim \mathcal{N}(14, 1) \\ \kappa / N_{\text{daytype}} &\sim \mathcal{D}_{N_{\text{daytype}}}(1, \dots, 1) \\ \sigma^2, \sigma_{s,0}^2, \sigma_{g,0}^2, \sigma_s^2, \sigma_g^2 &\sim \mathcal{IG}(10^{-2}, 10^{-2}) \end{aligned}$$

where $\mathcal{D}_d(\alpha_1, \dots, \alpha_d)$ is the Dirichlet distribution in $\mathbb{R}_+^d$ with parameter α (in particular $\mathcal{D}_d(1, \dots, 1)$ is the uniform distribution over the simplex of $\mathbb{R}_+^d$ defined by $\sum_{i=1}^d x_i = 1$).

Practical issue

We faced some technical issues running the MCMC estimation up until time $n_0 = 365$ since the Markov Chain outputs were not usable: even with a large burn-in period, the sample returned would not pass the diagnostic tests for the convergence of the empirical distribution towards the true target (see Gelman and Rubin, 1992, for example). For the initialisation via MCMC we thus separated the initial distribution into two parts, essentially isolating the dynamic on the variance of the random-walks, and proceeded as follows.

First we estimated the model as defined in (4.1) up until time $n_0 = 365$, using MCMC generic software such as described in Lunn et al. (2000); Plummer (2003), with the following modification

$$\begin{aligned} \sigma_{s,n} &= \sigma_{s,n-1} = \sigma_{s,*} \\ \sigma_{g,n} &= \sigma_{g,n-1} = \sigma_{g,*} \end{aligned}$$

with initialisation

$$\sigma_{s,*}^2, \sigma_{g,*}^2 \sim \mathcal{IG}(10^{-2}, 10^{-2}),$$

in essence removing the second layer in the dynamic from the model (since $\sigma_{s,n}$ and $\sigma_{s,n}$ are not allowed to vary with time anymore). This led to a posterior distribution on a diminished state, that we denote $\widetilde{\pi}_1(\widetilde{x}_{n_0-1}|y_{0:n_0-1})$. From there we completed this posterior distribution with an additional prior $\widetilde{\pi}_2$ on σ_s and σ_g to serve as an initialisation at for the full model at time n_0.

The initial distribution of the particle filter for the full model at time n_0 was thus of the form

$$\pi(x_{n_0-1}|y_{0:n_0-1}) \propto \widetilde{\pi}_1(\widetilde{x}_{n_0-1}|y_{0:n_0-1}) \times \widetilde{\pi}_2(\sigma_{s,n_0-1}, \sigma_{g,n_0-1})$$

with

$$\begin{aligned} \sigma_s^2 &\sim \mathcal{N}(\overline{m}_s, \overline{s}_s^2, \mathbb{R}_+^*) \\ \sigma_g^2 &\sim \mathcal{N}(\overline{m}_g, \overline{s}_g^2, \mathbb{R}_+^*) \end{aligned}$$

where $\overline{m}_s, \overline{m}_g, \overline{s}_s^2, \overline{s}_g^2$ were values chosen empirically based on $\widetilde{\pi}_1$.
For example, we chose m_s and m_g to be the standard deviations of the posterior MCMC estimated samples $(\mathbb{E}[\epsilon_1^s|y_{0:n_0}], \dots, \mathbb{E}[\epsilon_{n_0}^s|y_{0:n_0}])$ and $(\mathbb{E}[\epsilon_1^g|y_{0:n_0}], \dots, \mathbb{E}[\epsilon_{n_0}^g|y_{0:n_0}])$ respectively.

4.4 Predictions

Quality criterion

To assess the quality of the models we propose, we will mainly look at their respective predictive performances measured by Mean Absolute Percentage Error (MAPE). As a matter of fact, we are working with half-hourly data and we will model each half-hour independently from one another, a common choice given the type of data, thus leading to 48 separate daily model (see Section 4.2). Indexing the respective MAPE criteria of these models by the instant $i = 0, \dots, 47$ to which they are associated, and given their respective observations $y_{1,i}, \dots, y_{n,i}$, these models return 48 τ-day-ahead predictions defined as the expectations of the predictive distributions i.e. for $i = 0, \dots, 47$

$$\widehat{y}_{n+\tau,i} = \mathbb{E}[x_{n+\tau} | y_{0:n,i}]. \tag{4.2}$$

The corresponding predictive (with prediction horizon τ) MAPE criterion that we consider for these 48 models is defined, for $i = 0, \dots, 47$, by

$$\mathrm{MAPE}_i(\tau) = \frac{1}{n-\tau} \sum_{k=1}^{n-\tau} \left| \frac{\widehat{y}_{k+\tau,i} - y_{k+\tau,i}}{y_{k+\tau,i}} \right|$$

and we will most often aggregate the results as

$$\begin{aligned} \mathrm{MAPE}(\tau) &= \frac{1}{48} \sum_{i=0}^{47} \mathrm{MAPE}_i(\tau) \\ &= \frac{1}{48(n-\tau)} \sum_{i=0}^{47} \sum_{k=1}^{n-\tau} \left| \frac{\widehat{y}_{k+\tau,i} - y_{k+\tau,i}}{y_{k+\tau,i}} \right|. \end{aligned}$$

Operational predictions

We will also compare these models to the so-called operational prediction (available from 01/01/09 only, i.e. for the second half of our dataset only) i.e. the final prediction that was actually used by EDF. Note that the operational prediction $\mathrm{Pred}_{\mathrm{OP}}$ cannot be written as a prediction coming from a statistical model (even though we will sometimes abusively refer to it as the prediction from the operational model) : it combines manual adjustments and statistical models. $\mathrm{Pred}_{\mathrm{OP}}$ is computed as a 50%–50% mixture between the two predictions $\mathrm{Pred}_{\mathrm{DOAAT}}$ and $\mathrm{Pred}_{\mathrm{DCo}}$ that we briefly describe below.

The prediction $\mathrm{Pred}_{\mathrm{DOAAT}}$ is obtained as follows. A model similar to the one described in Bruhns et al. (2005), with an ARIMA part, is first used on a real-time estimated signal corresponding to the "France" perimeter. An estimated loss is then substracted from it, accounting for the customers within this perimeter that are not affiliated with EDF. A manual adjustment is finally applied in real-time. It is a "top-down" prediction in the sense that the "EDF" perimeter is approximated as a difference between the "France" perimeter and a "France but not EDF" perimeter.

The prediction $\mathrm{Pred}_{\mathrm{DCo}}$ is obtained as follows. Multiple models from Bruhns et al. (2005) are used upon consolidated signals (not available in real-time, only three weeks later) for sub-perimeters, the reunion of which is the "EDF" perimeter. The corresponding predictions are then added together before a manual adjustment is finally applied in real-time. It is a "bottom-up" prediction in the sense that the "EDF" perimeter is approximated as the sum of all its parts.

There are a number of differences between the dynamic predictions and the operational predictions. First of all, the operational predictions are computed using predicted temperatures (since the sequence of observed temperatures at the time of prediction is clearly not available) whereas the models that we consider (see Section 4.2) are based on the realised temperatures. The

operational predictions make use of a calendar that includes more daytypes and also benefit from high level expertise through the manual adjustments mentioned. But the biggest difference in nature between these predictions lies somewhere else: the dynamic predictions are made from one day to the next (with no intraday correction whatsoever, since we are basically considering 48 independent models), while the operational predictions are made from one half-hour to the next. Essentially the horizon of prediction for the dynamic models is $\tau = 1$ day $= 48$ half-hours whereas it is much smaller for $\text{Pred}_{\text{DOAAT}}$, since the new data get incorporated approximately every 8 half-hours (the computation of $\text{Pred}_{\text{DOAAT}}$ is based upon a real-time signal though, not consolidated data).

4.5 Results

Running the filter

For the estimation and prediction of the model, we used the Algorithm 3.10 with a total number of $M = 10^5$ particles. One time step (filtering and predicting the state with horizon $\tau = 1$, including 90% credible intervals) took approximately 1 second on a single core Intel(R) Xeon(R) E5410 (2.33GHz) for one of the 48 independent models, which is compatible with the goal of being able to predict the electricity load in an online manner. The execution time grew a bit larger and reached 3 seconds per iteration when the predictive horizon was set to $\tau = 5$. Note that providing credible intervals requires the use of a sorting algorithm, for example quicksort (see Hoare, 1962) with complexity $\mathrm{O}(M \log M)$ whereas Algorithm 3.10 has only complexity $\mathrm{O}(M)$. Quicker runtimes are thus obviously achievable if the computation of credible intervals is not needed.

Degeneracy

Before looking at the filtered or predicted distributions that we are most interested in, we actually have to assess whether the numerical results obtained are actually usable or not. If the degeneracy problem proved too strong along the estimation process, the estimated values indeed become questionable.

Figures 3, 4 and 5 show the evolution of the various criteria discussed in Section 3.2 throughout time for the model (4.1) at the instant 12:00. These criteria exhibit a seasonal behaviour (with a 1 year period), as the time series itself, showing that the particle filter is subject to a little more degeneracy during winter than during summer (the electricity load is indeed harder to predict, due to the influence of the temperature). Although the coefficient of variation $\mathrm{CV}(n)$ is only a rescaling of the effective sample size $\mathrm{ESS}(n)$ (see (3.3)), the outliers detected by the Algorithm 3.10 used are much easier to spot on Figure 5 than on Figure 3. Also observe that even if the entropy and the coefficient of variation approximate two different divergences (see Cornebise, 2009, for the details), the outliers are as easily spotted on Figures 4 and 5 and the behaviours of the two criteria are very similar : hence, using the entropy instead of the effective sample size (or the coefficient of variation, since they are interchangeable) to detect outliers in Algorithm 3.10 could be doable (after having developed a basic intuition of its scaling, in order to decide of a threshold) but would not change the results obtained in any major way.

Outliers

We show in Figure 6 the number of data that were automatically detected as outliers by the model for each instant (half-hour) of the day. Recall that, according to Algorithm 3.10, an outlier is detected whenever the effective sample size would have dropped below 0.1% of the actual sample size. The amount of outliers varies from one half-hour to the next because an observation flagged as an outlier at a given instant does not necessarily imply that the observation at the next instant will also be flagged. In particular, we observe that more outliers are detected during the day than

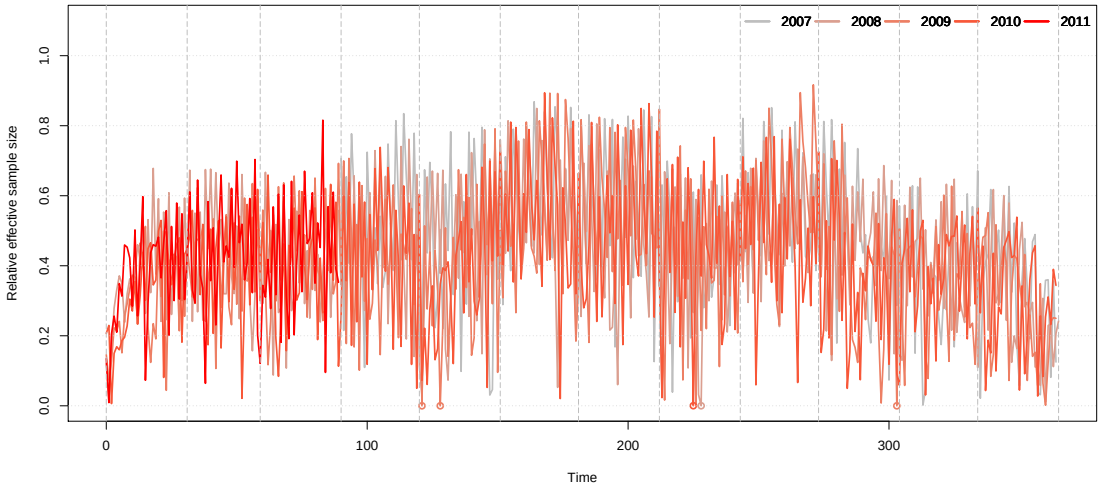

Figure 3: Relative effective sample size $\frac{\mathrm{ESS}(n)}{M}$ for the dynamic model (4.1) at 12:00 as a function of the day in the calendar. The saturation of the colour used increases with each year. Data detected as outliers are marked with a circle.

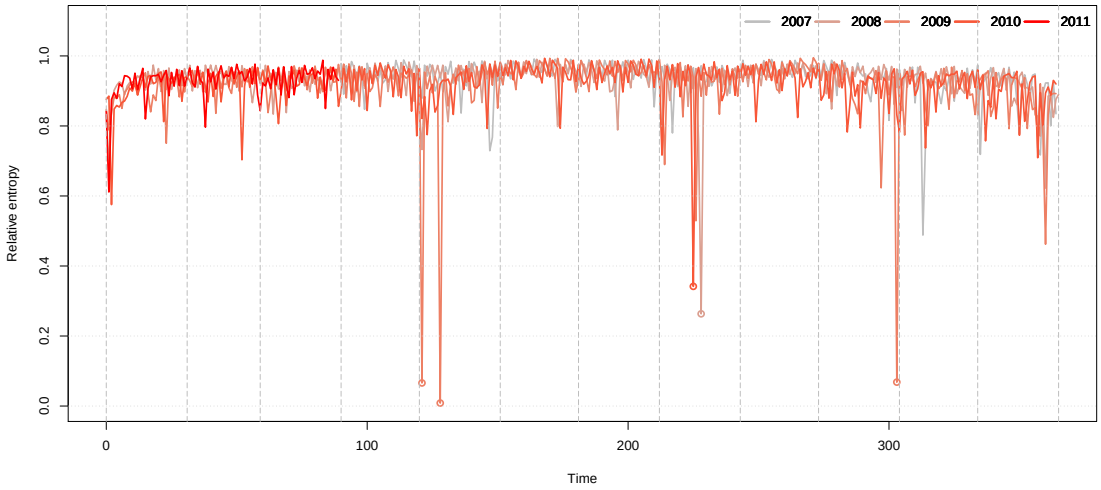

Figure 4: Relative entropy $\frac{\mathcal{E}(n)}{-\log M}$ for the dynamic model (4.1) at 12:00 as a function of the day in the calendar. The saturation of the colour used increases with each year. Data detected as outliers are marked with a circle.

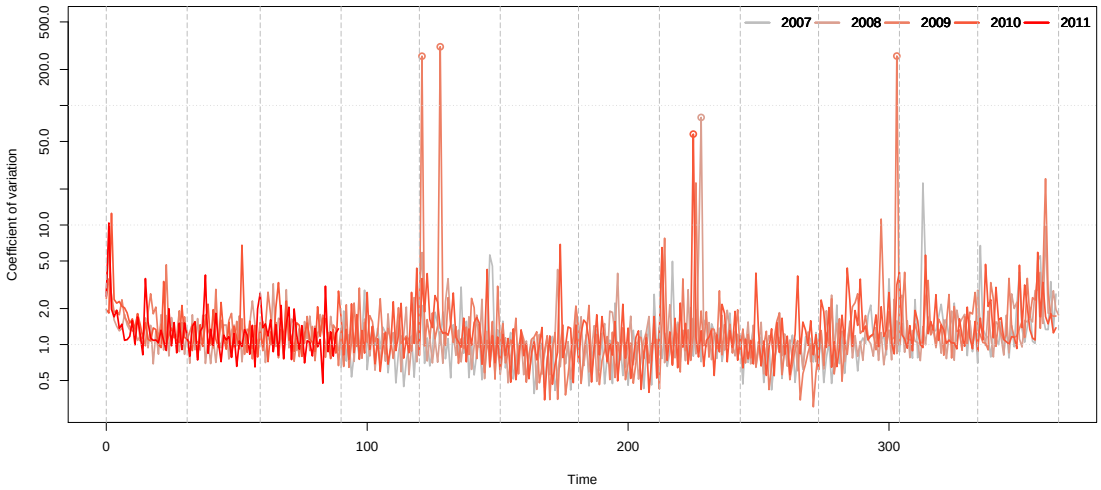

Figure 5: Coefficient of variation CV(n) for the dynamic model (4.1) at 12:00 as a function of the day in the calendar. The saturation of the colour used increases with each year. Data detected as outliers are marked with a circle. The ordinate axis is in log-scale.

during the night, which suggests that nighttime is slightly easier to predict than daytime (recall that outliers are essentially data that are badly predicted).

Figure 7 shows the number of outliers depending on the calendar. It allows us to pinpoint the times of the year at which these outliers are actually detected. The summer and winter holiday breaks, and the daylight saving time adjustments are easily spotted. Note that for these events, no prior information was available to the dynamic models. Some days before or after bank holidays are also flagged as outliers (05/02, 05/02, 11/10), even though the dynamic model benefits from some calendar information. This should not come as a surprise however: the daytype specification that we chose is rather poor compared to the calendar used for the operational predictions. A more refined calendar, involving specific daytypes, is likely to help turning these few outliers back into regular data, provided the initialisation of the particle filter is correctly done.

Table 2 summarises what is already guessable from Figures 2 and 7, i.e. that most of the (few) instants detected as outliers by the dynamic model are indeed bank-holidays.

instant	outlier	not outlier
bank-holiday	269 [5.60]	16627 [346.40]
not bank-holiday	38 [0.79]	53194 [1108.21]

Table 2: Classification of the instants for the dynamic model (4.1). The number given between square brackets is an equivalent of the number of instants in days (i.e. divided by 48).

Performance and instants

We show the overall predictive (horizon $\tau = 1$) performance of the dynamic model (4.1) against the operational model (OP) in Table 3, depending on whether bank-holidays were included in the calculations or not. The results shown in both cases aggregate the 48 models that were estimated independently from one another. Over the whole period of study, the operational predictions are

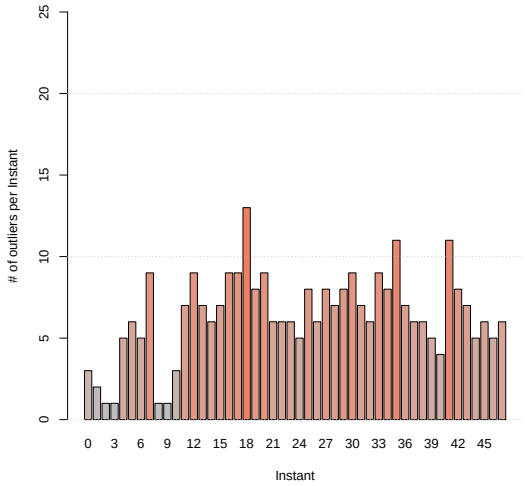

Figure 6: Number of outliers detected for each instant of the day by the dynamic model (4.1).

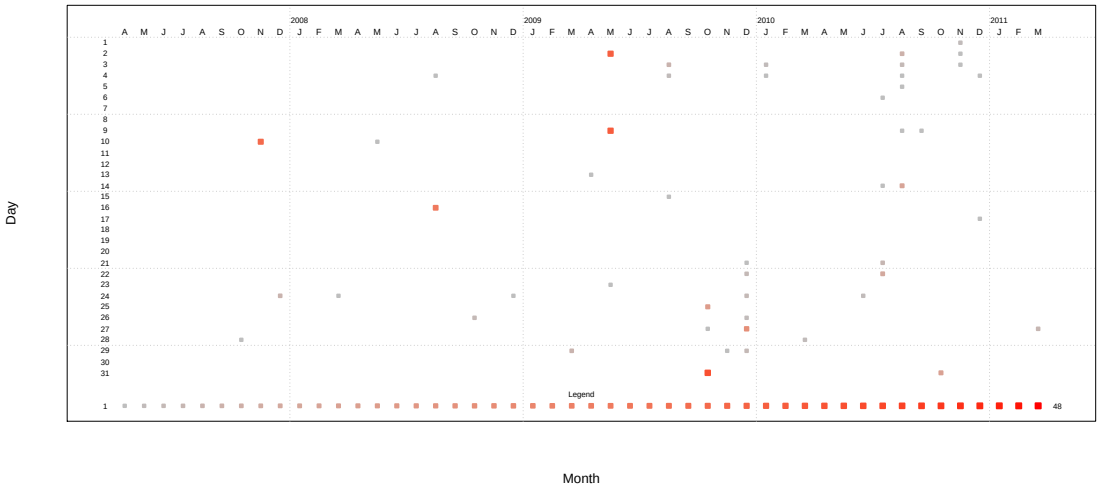

Figure 7: Number of outliers detected by the dynamic model (4.1) depending on the calendar from 04/01/2007 to 03/31/2011. Each column represents a month. The size of the point and the saturation of the colour used grow with the number of outliers, as indicated in the legend beneath.

better than the predictions provided by the dynamic models, but they also do benefit from more specific calendar information being used to compute them. When bank-holidays are removed from the calculations, the overall predictive quality of the dynamic models improves considerably as demonstrated by the results in Table 3.

	dynamic model	operational model
with bank-holidays	1.4342	1.2344
without bank-holidays	1.1712	1.2185

Table 3: Overall predictive (horizon $\tau = 1$) and MAPE (in %) for the dynamic model (4.1) and the operational model. The top row results include bank-holidays in the calculations, while the bottom row results do not.

In fact, looking at Figure 8 that represents the predictive MAPE of the dynamic model (dyn) and operational model (OP) averaged by instant, we are able to see that the dynamic model predicts the electricity load quite well when bank-holidays are not considered, challenging the operational model throughout the day, except during the morning ascent. The good predictive performance of the dynamic model on these days is somewhat surprising because the dynamic predictions, coming from 48 independent models, are made from one day to the next whereas the operational predictions include an ARIMA adjustment phase to take advantage of the most recent observations, and also benefit from manual adjustments.

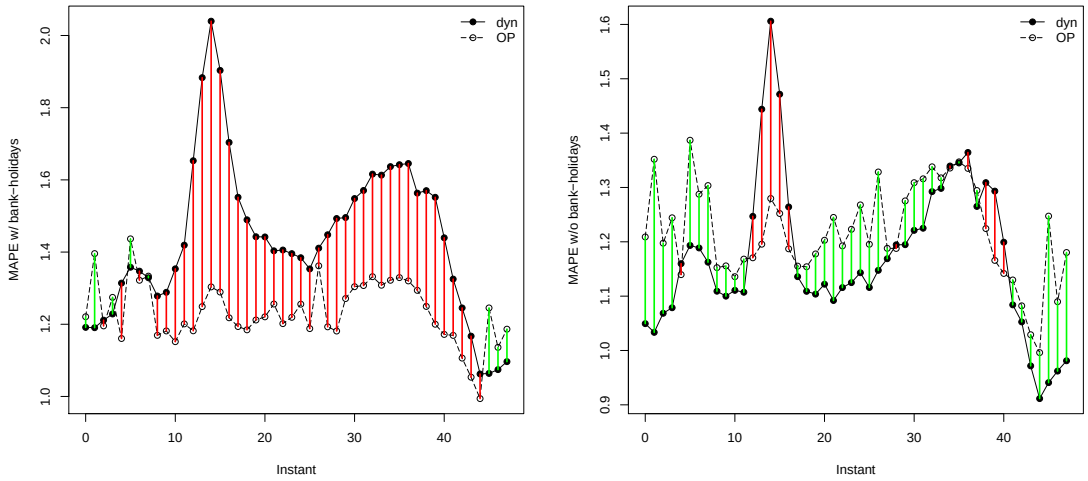

Figure 8: Predictive (horizon $\tau = 1$) and MAPE (in %) for the dynamic model (dyn) and the operational model (OP) for each of the 48 half-hours, including bank-holidays in the calculations (leftmost figure) and not including bank-holidays in the calculations (rightmost figure). The difference between the two models is coloured depending on its sign: green when the dynamic model is better than the operational model and red when not.

Performance and horizon

Since the operational predictions are sometimes required up to $\tau = 3$ days, we now investigate the predictive quality of our dynamic model as the horizon for prediction grows larger. Figure 9, given hereafter, displays the predictive MAPE for horizon $\tau = 1, \ldots, 5$, whether including bank-holidays in the calculations or not. It is clear that the predictive errors of the dynamic model increase with the horizon τ considered for the prediction, confirming that it is primarily meant for short-term forecasts and not long-term forecasts.

Another consequence of increasing the prediction's horizon is that the credible intervals obtained around the predictions also tend to grow larger on average as can be observed in Table 4. An illustration of the credible intervals returned by the dynamic models is given in Figure 10 where the electricity load is predicted over 48 consecutive instants via the dynamic model (4.1). The predictions clearly improve over time as the model takes more and more recent information into account: the one-day-ahead predictions about 12/30/2010 provided on 12/29/2010 are much more accurate than the five-days-ahead predictions (of the same day) that were computed on 12/25/2010. Figure 10 also makes it clear that the credible intervals obtained for a predictive horizon $\tau = 1$ are narrower compared to those obtained for a predictive horizon $\tau = 5$ (but note that their lengths vary over time).

	$\tau = 1$	$\tau = 2$	$\tau = 3$	$\tau = 4$	$\tau = 5$
$\widehat{\lambda}_{90\%}(x_{n+\tau})$	2746.3	3721.1	4505.6	5191.6	5815.3
$\widehat{\lambda}_{90\%}(y_{n+\tau})$	3036.1	3947.7	4696.5	5358.6	5964.9

Table 4: Mean length (in MW) of the symmetric 90% credible intervals (CI) around the predicted states $\widehat{x}_{n+\tau}$ and around the predicted observations $\widehat{y}_{n+\tau}$ of the dynamic model (4.1), for $\tau = 1, \ldots, 5$.

	$\tau = 1$	$\tau = 2$	$\tau = 3$	$\tau = 4$	$\tau = 5$
$\widehat{\chi}_{90\%}(\widehat{x}_{n+\tau})$	89.569	90.442	92.385	93.479	94.168
$\widehat{\chi}_{90\%}(\widehat{y}_{n+\tau})$	92.531	92.501	93.773	94.472	94.882

Table 5: Empirical coverage (in %) of the symmetric 90% credible intervals (CI) around the predicted states $\widehat{x}_{n+\tau}$ and around the predicted observations $\widehat{y}_{n+\tau}$ of the dynamic model (4.1), for $\tau = 1, \ldots, 5$.

The empirical coverages of the symmetric 90% credible intervals around the predicted states and observations are given in Table 5. These values were computed as the ratio between the number of instants for which the observations fell inside the interval, and the total number of instants. Note that if the observations were mutually independent outcomes of the same random variable (which they are not in our situation because of the exogenous variables temperature and calendar), this ratio would theoretically approximate the true rate of coverage i.e. 90%. Even so, the empirical coverage computed seems, somewhat reassuringly, to agree with the expected rate.

Filtered weather parts

The Figure 11 shows the filtered heating and cooling parts of the dynamic model (4.1). It seems to be piecewise linear with regard to the temperature variables upon which it depends, with a threshold that depends on the instant considered. The heating part however is not modelled as

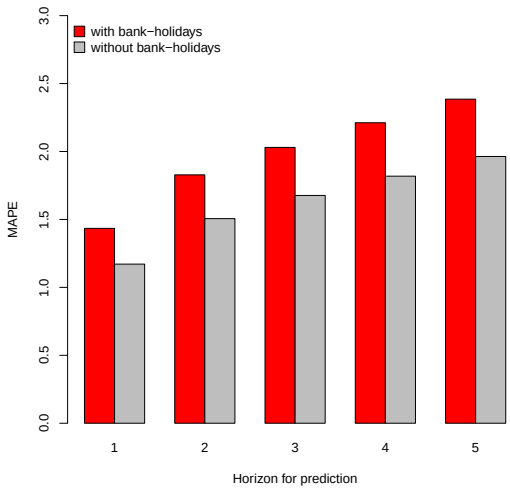

Figure 9: Predictive MAPE of the dynamic model (4.1) for $\tau = 1, \ldots, 5$, including bank-holidays in the calculations (leftmost bars) and not including bank-holidays in the calculations (rightmost bars).

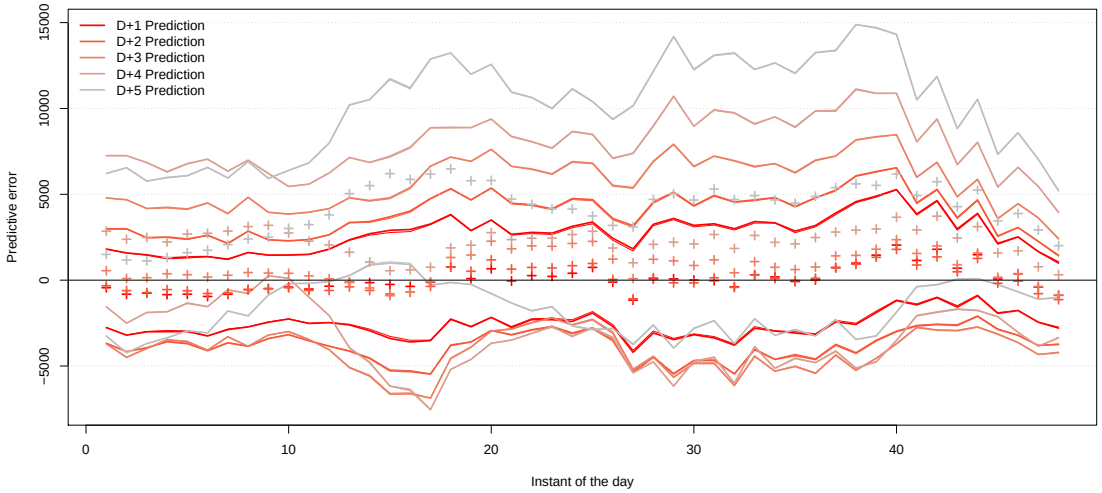

Figure 10: Predictive errors (predictive mean minus true value) of the dynamic model (4.1) for the observations on 12/30/2010 (48 half-hours) with $\tau = 1, \ldots, 5$. The horizontal black line marks the true load (no predictive error), and crosses mark the various predictive errors with their respective credible intervals in solid lines. The more recent the predictions are (i.e. the smaller τ is), the more saturated the colour used is: D+1 Prediction is the most recent prediction (it was made 1 day before) while D+5 is the oldest prediction (it was made 5 days before).

such since the heating gradient is chosen non constant in the dynamic model. It is thus a bit of a surprise to find this familiar piecewise linear shape for the heating part, even though it is quite common for non dynamic models (see Bruhns et al., 2005, for example).

As in Dordonnat et al. (2008), the heating gradient of the dynamic model appeared to be stronger in winter and slightly weaker over mid-seasons (note that the behaviour of the heating gradient over summer is of little practical importance: while it is true that it cannot be observed accurately at that time, it also has no direct impact on the quality of the model since this is precisely the period over which the heating part of the model vanishes). Note that, on the contrary, the variance of the heating gradient appeared to be larger during summer (with no information available) than during winter.

Filtered seasonal part

Even though we do not display it here, let us mention that the filtered seasonal part x_n^{season} of the dynamic model exhibits a 1-year period with weekly cycles. Around the main periodic pattern, variations occur : more so over the winter period, for which the seasonal part is obviously not so well defined, than over the summer period. Indeed, during summer the seasonal part is the only active dynamic part of the model, while during winter the heating part also plays an important role : the estimated values of both parts over winter are thus to be interpreted with caution. Still, the filtered seasonal part seems to react correctly to the summer and winter holiday breaks (as we will outline in the next Section), although no particular information was used to flag these time windows for the model.

Because EDF customers now represents a fraction only of the French customers population (instead of the whole), the perimeter of the data varies over time due to customers departures or arrivals (but taking into account that EDF and France perimeters were actually identical until a few years ago, departures are a bit more likely). As a matter of fact, the filtered seasonal part also shows successive yearly drops from 2008 and onwards, which correspond to the financial crisis that arose in late 2008 (and that impacted the French electricity load), or planned customers' departures.

Summer break

Since holiday breaks are among the most toughest times of the year for predictions, we investigate the behaviour of the dynamic models over the summer break to show how the models cope with the difficulty.

Evolution of the dynamics The Figure 12 shows the filtered mean of both s_n and $\sigma_{s,n}$, that rules the dynamic of s_n within the dynamic model (4.1). As can be seen on the Figure 12, the model is able to filter out the summer break effectively : to allow for the sharp drop of s_n during August, the standard deviation of its dynamic $\sigma_{s,n}$ suddenly grows (becoming twice as large as usual), reflecting the brusque increase of variability of the signal over a short period of time. The model also deals with the winter break in a similar manner.

We have already discussed the behaviour of the heating gradient over summer : during summer the model logically loses track of anything related to the heating part, which leads to artificially increased values of $\sigma_{g,n}$.

The reasons behind the increased values of $\sigma_{s,n}$ and $\sigma_{g,n}$ during the summer break are hence entirely different. Whereas $\sigma_{s,n}$ grows to allow the model to fit data that do not match the current state, the growth of $\sigma_{g,n}$ merely reflects the lack of cold temperatures that would help estimate any of the coefficients related to the heating part of the dynamic model.

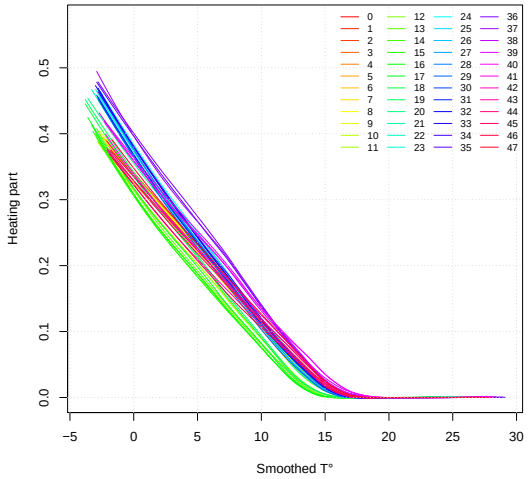

Figure 11: Estimated filtered heating part of the dynamic model (4.1) against the smoothed temperature T_n^{heat} given to the model. 48 distinct colours are used, one for each of the 48 half-hours. The estimation was done via the loess function in R considering the filtered mean of the heating part against both temperatures. Only the relative heating part is shown here, i.e. the heating part divided by the maximum load observed over the whole period.

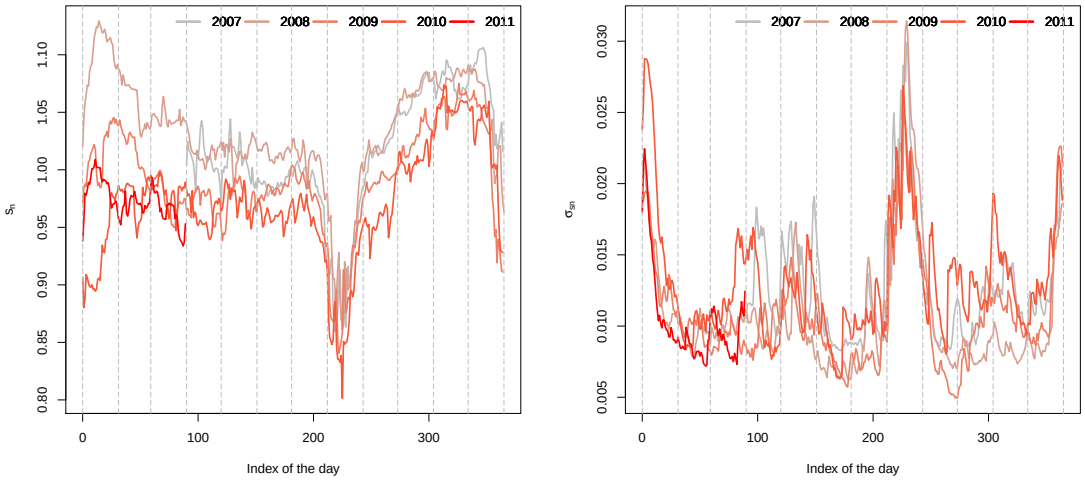

Figure 12: Mean of the filtered coefficients s_n (left) and $\sigma_{s,n}$ (right) of the dynamic model (4.1) averaged over 48 half-hours, as functions of the day in the calendar. The saturation of the colour used increases with each year. Only the relative filtered means of s_n and $\sigma_{s,n}$ are shown here, i.e. the means divided by the mean of s_n over the whole period.

Predictive errors Though no information is provided about the summer break (a succession of breaks mostly occurring on Mondays), we already saw that the dynamic model is able to estimate the electricity load rather correctly given the peculiar circumstances.

A possible way to improve the quality of the forecasts for the days where breaks occur would be to taylor the transition density of one state to the next specifically for them. This requires much expertise in practise because the way the load is affected by the summer break also depends on the calendar configuration: one could for instance introduce adequately modified specifications (interventions) such as

$$s_{n*} = s_{n*-1} - \mu_{n^*} + \epsilon^s_{n^*}$$

into the model where $\mu_{n^*} \in \mathbb{R}_+$ is the drop in load expected to happen at time n^*.

Comparison with a linear Gaussian state space model

A dynamic model was proposed and studied by Dordonnat et al. (2008) to model a similar electricity load series (at the French national perimeter). Their model fit in the multivariate linear Gaussian state space models framework which allowed for the use of Kalman filtering and associated techniques (see Durbin and Koopman, 2001). It is actually quite a complex and rich model, compared to our own, and includes multiple regressions, some coefficients of which are allowed to vary over time : a truncated Fourier series is used to model the seasonality of the signal as in Bruhns et al. (2005) in conjunction with a stochastic trend. Local trends are also included to model the holiday breaks, and a calendar with various specific daytypes is used. Heating and cooling parts are defined as well, using fixed threshold values (15°C and 18°C) as well as fixed smoothing parameters (fixed to $\vartheta = 0.98$), and are thus very similar to the ones we use, although the heating part relied upon the use of two heating gradients (the first corresponding to the raw temperature, the other to the difference between smoothed and raw temperatures). The model was estimated using national data from 09/01/1995 to 08/30/2004, and its predictive quality was assessed from 09/01/2003 to 08/30/2004 only.

Let us first mention that the performances reported by Dordonnat et al. (2008) for their model are in accord with ours, with a one-day-ahead predictive MAPE varying around 1.30% across the 24 hours considered, and larger errors during the weekends or holiday breaks. They also found the quality of the forecasts obtained to be degrading with the predictive horizon, just as we did, and at a similar rate. Finally, the behaviour of the heating gradient that we reported corroborates the behaviour of the heating gradients found in Dordonnat et al. (2008) (with this difference that they used a smoothing approach for the signal extraction, whereas we used a filtering approach).

Still, the dynamic model (4.1) that we propose is much simpler, most notably where the seasonality part is concerned: our model only includes 9 daytypes and at most 2 temperatures, i.e. 10 random effects whereas the model described in Dordonnat et al. (2008) made use of more than 30 random effects. Arguably, the estimation time of our model via a particle filter takes more time than running a Kalman filter, but particle filters naturally allows for more flexibility in the definition of the model (including non-linear non-Gaussian model). Most importantly, the Algorithm 3.10 that we implemented for the estimation of our models automatically treats bank-holidays instants as missing data when Dordonnat et al. (2008) explicitly and manually had to declare which data had to be considered as missing data, so as not to throw the model off. Also note that even though the model studied in Dordonnat et al. (2008) was more complex, the predictive MAPE they obtained for non regular daytypes exceeded 5% at 09:00AM and 12:00PM the two instants they focused on while the dynamic model (4.1) had an averaged predictive MAPE of 3.34% for non regular daytypes (but once again keep in mind that the datasets used for their experiments and ours were different which may possibly explain part of the observed difference).

References

Andrieu, C., de Freitas, N., and Doucet, A. (1999). Sequential MCMC for Bayesian Model Selection. In *IEEE Higher Order Statistics Workshop, Ceasarea*, pages 130–134.

Andrieu, C., Doucet, A., and Holenstein, R. (2010). Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society: Series B*, 72(3):269–342.

Bruhns, A., Deurveilher, G., and Roy, J. (2005). A Non-Linear Regression Model for Mid-Term Load Forecasting and Improvements in Seasonnality. *Proceedings of the 15th Power Systems Computation Conference 2005, Liege Belgium*.

Cappé, O., Moulines, E., and Ryden, T. (2010). *Inference in Hidden Markov Models (Springer Series in Statistics)*. Springer.

Carpenter, J., Clifford, P., and Fearnhead, P. (1999). Improved particle filter for nonlinear problems. *Radar, Sonar and Navigation, IEE Proceedings*, 146(1):2–7.

Casarin, R. and Marin, J.-M. (2009). Online data processing: comparison of Bayesian regularized particle filters. *Electron. J. Statist.*, 3:239–258.

Chen, Z. (2003). Bayesian Filtering : From Kalman Filters to Particles Filters, and Beyond.

Chopin, N., Jacob, P. E., and Papaspiliopoulos, O. (2012). SMC^2: an efficient algorithm for sequential analysis of state-space models. *Journal of the Royal Statistical Society: Series B*.

Cornebise, J. (2009). *Méthodes de Monte Carlo Séquentielles Adaptatives*. PhD thesis, Université Pierre et Marie Curie.

DasGupta, A. (2008). *Asymptotic Theory of Statistics and Probability*. Springer Texts in Statistics. Springer, 1st edition.

Dordonnat, V. (2009). *State-space modelling for high frequency data*. PhD thesis, Vrije Universiteit Amsterdam.

Dordonnat, V., Koopman, S., Ooms, M., Dessertaine, A., and Collet, J. (2008). An Hourly Periodic State Space Model for Modelling French National Electricity Load. *International Journal of Forecasting*, 24(4):566–587.

Douc, R. and Cappe, O. (2005). Comparison of resampling schemes for particle filtering. In *Image and Signal Processing and Analysis, 2005. ISPA 2005. Proceedings of the 4th International Symposium on*, pages 64–69. IEEE.

Doucet, A. (1998). On sequential Simulation-Based methods for bayesian filtering. Technical Report CUED/F-INFENG/TR. 310, Cambridge University Department of Engineering.

Doucet, A., De Freitas, N., and Gordon, N. (2001). *Sequential Monte Carlo Methods in Practice (Statistics for Engineering and Information Science)*. Springer, 1st edition.

Doucet, A., Godsill, S., and Andrieu, C. (2000). On Sequential Monte Carlo Sampling Methods for Bayesian Filtering. *Statistics and Computing*, 10:197–208.

Doucet, A. and Johansen, A. M. (2011). A tutorial on particle filtering and smoothing: fifteen years later.

Durbin, S. and Koopman, S. (2001). *Series Analysis by State Space Methods*. Oxford Statistical Science Series.

Gelman, A. and Rubin, D. B. (1992). Inference from Iterative Simulation Using Multiple Sequences. *Statistical Science*, 7(4):457–472.

Geweke, J. (1989). Bayesian Inference in Econometric Models Using Monte Carlo Integration. *Econometrica*, 57(6):1317–1339.

Gilks, W. R. and Berzuini, C. (2001). Following a moving target – Monte Carlo inference for dynamic Bayesian models. *Journal of the Royal Statistical Society: Series B*, pages 127–146.

Gordon, N. J., Salmond, D. J., and Smith, A. F. M. (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *Radar and Signal Processing, IEE Proceedings F*, 140(2):107–113.

Gustafsson, F., Gunnarsson, F., Bergman, N., Forssell, U., Jansson, J., Karlsson, R., and Nordlund, P. J. (2002). Particle filters for positioning, navigation, and tracking. *IEEE Transactions on Signal Processing*, 50(2):425–437.

Higuchi, T. (2001). Self-organizing time series model. In *Sequential Monte Carlo Methods in Practice*, pages 429–444. Springer-Verlag.

Hoare, C. A. R. (1962). Quicksort. *The Computer Journal*, 5(1):10–16.

Hu, X.-L., Schön, T. B., and Ljung, L. (2008). A basic convergence result for particle filtering. *IEEE Transactions on Signal Processing*, 56(4):1337–1348.

Hu, X.-L., Schön, T. B., and Ljung, L. (2011). A general convergence result for particle filtering. *IEEE Transactions on Signal Processing*, 59(7):3424–3429.

Johansen, A., Doucet, A., and Davy, M. (2008). Particle methods for maximum likelihood estimation in latent variable models. *Statistics and Computing*, 18(1):47–57.

Kantas, N., Doucet, A., Singh, S., and Maciejowski, J. M. (2009). Overview of Sequential Monte Carlo methods for parameter estimation on general state space models. In *15th IFAC Symposium on System Identification (SYSID), Saint-Malo, France*. (invited paper).

Karlsson, R. (2005). *Particle Filtering for Positioning and Tracking Applications*. Linköping Studies in Science and Technology. Thesis No 924, Linköping Universitet.

Kitagawa, G. (1996). Monte carlo filter and smoother for Non-Gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics*, 5(1):1–25.

Kitagawa, G. (1998). A Self-Organizing State-Space Model. *Journal of the American Statistical Association*, 93(443):1203–1215.

Kong, A., Liu, J. S., and Wong, W. H. (1994). Sequential imputations and bayesian missing data problems. *Journal of the American Statistical Association*, 89(425):278–288.

Launay, T., Philippe, A., and Lamarche, S. (2012a). Consistency of the posterior distribution and MLE for piecewise linear regression. *Electron. J. Statist.*, 6:1307–1357.

Launay, T., Philippe, A., and Lamarche, S. (2012b). Construction of an informative hierarchical prior distribution. Application to electricity load forecasting. *Preprint*. arXiv:1109.4533.

Lee, J.-Y., Payandeh, S., and Trajković, L. (2010). The internet-based teleoperation: Motion and force predictions using the particle filter method. *ASME Conference Proceedings*, 2010(44458):765–771.

Liu, J. and West, M. (2001). Combined parameter and state estimation in simulation-based filtering. In Freitas, D. and Gordon, N. J., editors, *Sequential Monte Carlo Methods in Practice*. Springer-Verlag, New York.

Liu, J. S. (2008). *Monte Carlo Strategies in Scientific Computing*. Springer, corrected edition.

Liu, J. S. and Chen, R. (1995). Blind Deconvolution via Sequential Imputations. *Journal of the American Statistical Association*, 90(430).

Liu, J. S. and Chen, R. (1998). Sequential Monte Carlo Methods for Dynamic Systems. *Journal of the American Statistical Association*, 93:1032–1044.

Lunn, D. J., Thomas, A., Best, N., and Spiegelhalter, D. (2000). WinBUGS - A Bayesian modelling framework: Concepts, structure, and extensibility. *Statistics and Computing*, 10(4):325–337.

Marin, J.-M. and Robert, C. (2007). *Bayesian Core : A Practical Approach to Computational Bayesian Statistics*. Springer.

Oudjane, N. (2000). *Stabilité et approximations particulaires en filtrage non linéaire, Application au pistage*. PhD thesis, Université de Rennes 1.

Pitt, M. K. and Shephard, N. (1999). Filtering via simulation: Auxiliary particle filters. *Journal of the American Statistical Association*, 94(446):590–599.

Plummer, M. (2003). JAGS: A program for analysis of bayesian graphical models using gibbs sampling. In *Proceedings of the 3rd International Workshop on Distributed Statistical Computing*.

Robert, C. P. (1996). *Methodes de Monte Carlo par chaines de Markov*. Economica.

Robert, C. P. and Casella, G. (2004). *Monte Carlo Statistical Methods*. Springer, 2nd edition.

Rossi, V. (2004). *Filtrage non linéaire par noyaux de convolution, Application à un procédé de dépollution biologique*. PhD thesis, Ecole Nationale Supérieure Agronomique de Montpellier.

Rui, Y. and Chen, Y. (2001). Better proposal distributions: Object tracking using unscented particle filter. In *2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2001), with CD-ROM, 8-14 December 2001, Kauai, HI, USA*, pages 786–793. IEEE Computer Society.

Silverman, B. W. (1986). *Density Estimation for Statistics and Data Analysis (Chapman & Hall/CRC Monographs on Statistics & Applied Probability)*. Chapman and Hall/CRC, 1st edition.

van der Merwe, R., de Freitas, N., Doucet, A., and Wan, E. (2001). The Unscented Particle Filter. In *Advances in Neural Information Processing Systems 13*.

Vavoulis, D. V., Straub, V. A., Aston, J. A. D., and Feng, J. (2012). A Self-Organizing State-Space-Model Approach for Parameter Estimation in Hodgkin-Huxley-Type Models of Single Neurons. *PLoS Comput Biol*, 8(3):e1002401.

Wan, E. A. and van der Merwe, R. (2000). The unscented Kalman filter for nonlinear estimation. In *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*, pages 153–158. IEEE.

Whiteley, N. and Johansen, A. M. (2011). Auxiliary particle filtering : recent developments. In Barber, D., Cemgil, A. T., and Chiappa, S., editors, *Bayesian time series models*. Cambridge University Press, Cambridge.